國立交通大學

資訊科學與工程研究所

碩士論文

「可視秘密碎片」馬賽克畫—一種新的藝術與其在資訊隱藏上的應用

Secret-fragment-visible Mosaic—a New Art and Its Applications to Information Hiding

研 究 生：賴怡臻

指導教授：蔡文祥　教授

中華民國九十九年六月

「可視秘密碎片」馬賽克畫—一種新的藝術與其在資訊隱藏上的應用

Secret-fragment-visible Mosaic—a New Art and Its Applications to Information Hiding

研 究 生：賴怡臻　　Student: I-Jen Lai

指導教授：蔡文祥　　Advisor: Prof. Wen-Hsiang Tsai

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

In partial Fulfillment of the Requirements

For the Degree of

Master

In

Computer Science

June 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年六月

「可視秘密碎片」馬賽克畫—一種新的藝術與其在資訊隱藏上的應用

研究生：賴怡臻　　指導教授：蔡文祥 博士

國立交通大學資訊科學與工程研究所

摘要

在本論文中，我們提出了一個新的藝術畫 －「可視秘密碎片馬賽克畫」，並應用在這種藝術畫，發展出三種不同的資訊隱藏技術 － 秘密傳輸、影像隱匿術及秘密分享。這種藝術畫是將一張「秘密影像」切割成許多正方形碎片，並用這些碎片當成元件，組成起來的新型式馬賽克畫。首先，我們針對人類視覺對顏色的敏感度，提出了一表示影像色彩分佈的公式，並用它從資料庫中篩選出與秘密影像最相像的候選圖片，來當作製作馬賽克畫的「目標影像」。有了秘密影像以及目標影像之後，我們利用本研究所提出的一個貪婪演算法，將秘密影像的碎片逐一嵌合到目標影像上。另外，當資料庫中的圖片數量不足時，選出的目標影像的色彩分佈會與秘密影像不同，導致製成的馬賽克畫和目標影像相差過遠。為此我們也提出了一個彌補這種情況的方法。

在秘密傳輸方面，我們是在製作可視秘密碎片馬賽克畫的時候，根據「在同一長方圖容器中(histogram bin)的秘密影像碎片會擁有相似的顏色」的這項特性，交換秘密影像碎片所對應到的目標影像區塊的標籤，來達到藏入秘密訊息的效果。在影像隱匿術方面，我們是將秘密文件轉換成灰階影像，並用它來製成一新的馬賽克畫，藉此達到隱藏秘密文件的效果，並根據灰階值提出另一個特徵值的計算方法，用以提高馬賽克畫製作的速度。最後，在秘密分享方面，我們是將一張秘密影像分成多張可視秘密碎片馬賽克畫，並將秘密影像的碎片平均分散在各個目標影像裡。

除了上述的方法外，我們還提出了幾個增加安全性的方法，確保藏入的秘密資訊不被駭客發現並攻擊。以上的方法皆有實驗結果證明它們在視覺方面的良好成效，以及在資訊隱藏技術上的可行性。

Secret-fragment-visible Mosaic—a New Art and Its Applications to Information Hiding

Student: I-Jen Lai Advisor: Wen-Hsiang Tsai
Institute of Computer Science and Engineering
National Chiao Tung University

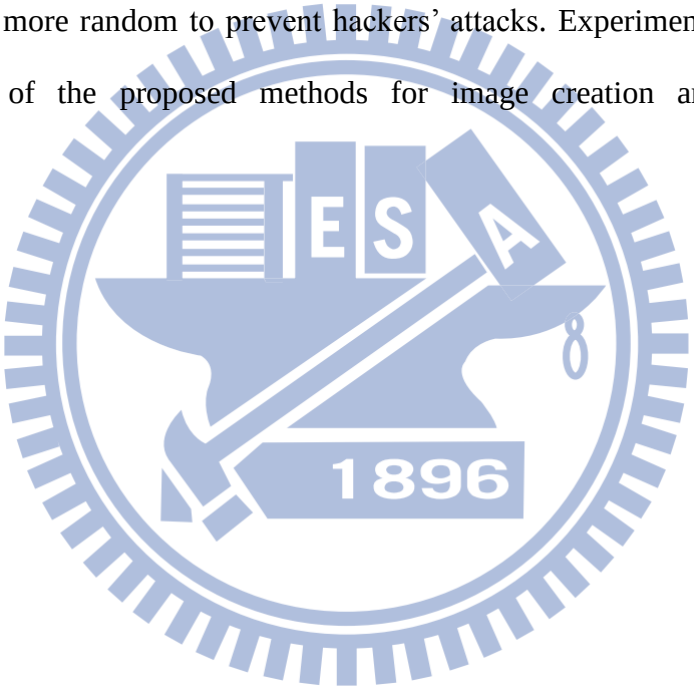
ABSTRACT

In this study, a new type of art image is created, which is called secret-fragment-visible mosaic image. And this kind of new mosaic image is used for three applications of data hiding, namely, covert communication, steganography, and secret sharing.

First, the newly-proposed secret-fragment-visible mosaic image is created to be composed of rectangular-shaped fragments which come from division of a secret image. A new 1-D *h*-colorscale is proposed to represent the color distribution of an image based on the color feeling of human vision. To create a secret-fragment-visible mosaic image, a new image similarity measure based on the *h*-colorscale is proposed, and the most similar candidate image from an image database is selected accordingly as a target image. Then, a greedy algorithm is adopted to fit every tile image in the secret image into an appropriate block in the target image. Furthermore, to solve the problem of using an insufficiently-large database, a remedy method by enlarging the size of a target image is also proposed. Secondly, for covert communication, based on the fact that tile images which are in an identical histogram bin have similar colors, the tile images in an identical histogram bin are reordered, or equivalently, the relative positions of the tile images are switched, to embed secret message bits imperceptibly. Third, for image steganography, a grayscale secret-fragment-visible mosaic image is

created from a grayscale image of a secret document, yielding a steganographic effect of hiding the secret document into the mosaic image, though visibly. The selection of the most similar target image from a database is based on a newly-proposed *k*-feature of the grayscale value, which speeds up the image creation process. Finally, for secret sharing, a secret image is shared to yield multiple secret-fragment-visible mosaic images. In order to disperse tile images evenly in selected target images, the target images take turns randomly to pick appropriate tile images.

In addition, various security enhancement measures were proposed to make the embedded data more random to prevent hackers' attacks. Experimental results show the feasibility of the proposed methods for image creation and data hiding applications.

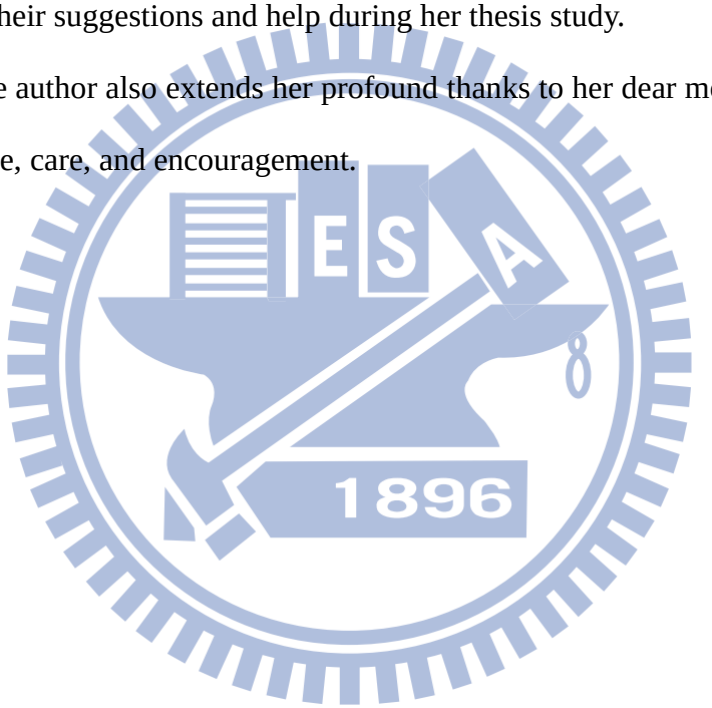


ACKNOWLEDGEMENTS

The author is in hearty appreciation of the continuous guidance, discussions, and support from her advisor, Dr. Wen-Hsiang Tsai, not only in the development of this thesis, but also in every aspect of her personal growth.

Appreciation is also given to the colleagues of the Computer Vision Laboratory in the Institute of Computer Science and Engineering at National Chiao Tung University for their suggestions and help during her thesis study.

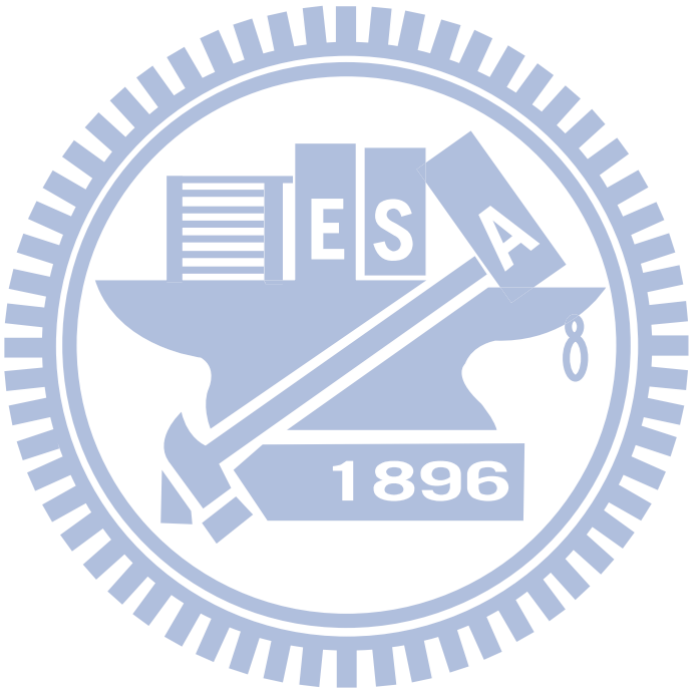
Finally, the author also extends her profound thanks to her dear mom and dad for their lasting love, care, and encouragement.



CONTENTS

LIST OF FIGURES

LIST OF TABLES



Chapter 1
Introduction

1.1 Motivation and Background

1.1.1 Research Motivation

Over the last decade, the Internet has become more and more indispensable to people's daily life. Many social network services have been built on the Internet in recent years. Therefore, people can communicate with friends and share their information, such as interesting videos or images, through the services. Part of the information, such as intelligence data, web bank account passwords, and so on, is private and should be protected carefully. When the owners of such messages want to deliver them to other people, hackers may easily steal or tamper with the data contents. Accordingly, information hiding becomes an important issue nowadays.

On the other hand, artworks often arouse interests of people in daily life, especially artistic pictures which attract people's visual attention. Recently more and more visual arts are produced by computers, creating a new type of art, called *computer art*. An example is *mosaic image*, though it is a form of art existing for thousands of years, dating back to the days of Rome and Greece in the western world.

Many information hiding techniques have been proposed in the past decade. Some of them were proposed via the use of images. The images, as carriers of information, were used just for the sole purpose of hiding information. If the images could be artworks like mosaic images, more attention of the people who receive them

will be attracted to appreciate their artistic contents, thus ignoring any other non-artistic function of the image like information hiding.

Therefore, unlike traditional information hiding techniques, in this study we try to combine data hiding and art image creation techniques for the purpose of information hiding. We will try in the first place to design new techniques for creating new types of computer art *in image form*. We will also investigate new techniques of embedding secret messages into the newly-designed computer art images during the creation process of them, yielding new ways of hiding information. With such disguise of the art image, illicit people who intend to thieve the hidden information will tend to believe that the image is only an artistic production and so ignore the information embedded in it, as mentioned previously.

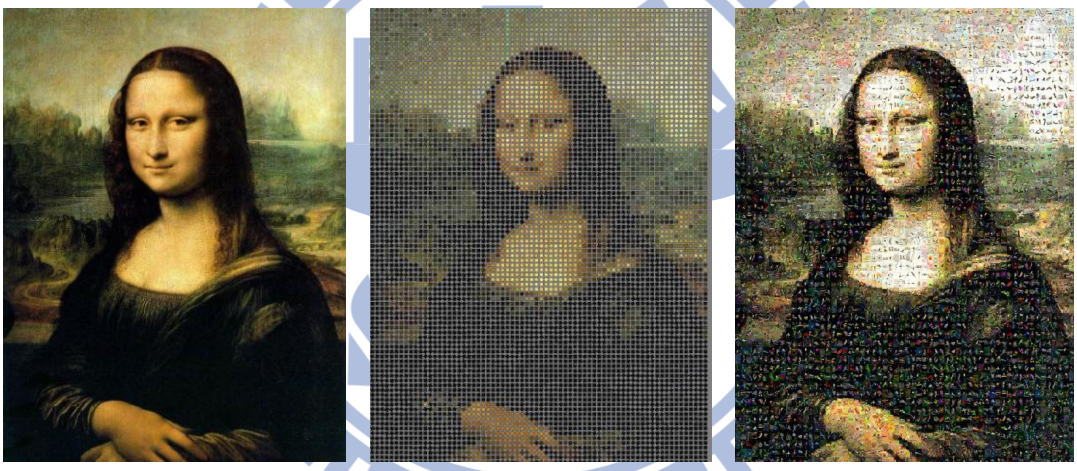
Furthermore, it is desired to use the designed new artworks as carriers for use in as many kinds of information hiding technique as possible. That is, we will try to design new information hiding techniques for various purposes via the newly-created computer art images. Specifically, via the use of the new type of image we will try to propose new methods for image steganography, covert communication, and secret sharing, all being different types of information hiding. These methods should all make good use of the characteristics of the new type of image we are going to design.

1.1.2 Introduction to Mosaic Images

Mosaics are the art of creating works, each being composed of small pieces, such as stone, glass, or other materials. The history of mosaics goes back to 4000 years ago or more, with the use of terracotta cones to conduct various types of decorations. With the rise of Christianity, mosaics were used to decorate the windows or ceilings of churches. Nowadays, it has turned into a common type of decoration for use in

modern houses, public space, advertisements, books, and so on.

In recent years, many researches have been conducted on the creation of different kinds of mosaic image. Traditional mosaic images are composed of a large number of small images, called *tile images*. To create a mosaic image, a user must choose an image first, called the *source image*. Then, the source image is divided into numerous rectangular pieces, each of which, called a *target image*, is replaced by a tile image similar in content. Consequently, while we see a mosaic image from a distance, as a whole it will look like its source image — an effect of a human vision property. An example is shown in Figure 1.1.



(a) (b) (c)

Figure 1.1 The Mona Lisa. (a) The original image. (b) A tile mosaic image [1]. (c) A photographic mosaic image [2].

1.2 Introduction to a New Type of Mosaic Image Proposed in This Study — Secret-fragment-visible Mosaic Images

1.2.1 New Idea of Secret-fragment-visible Mosaic Image

A new type of art image which contains small fragments of a source image is proposed in this study. The idea was inspired by the sliding puzzle like that shown in Figure 1.2. The sliding puzzle challenges a player to slide randomly-placed block pieces of a picture (like Figure 1.2(a)) with a certain way to rebuild the original picture content (like Figure 1.2(b)). Note that the randomly-placed block pieces as a whole show a *meaningless* picture.

We extend in this study the idea of slide puzzle to create a new type of art image, called *secret-fragment-visible mosaic image*, which is composed of the fragments of a source image. Different from the sliding puzzle which, when randomized, looks meaningless as mentioned previously, the randomly-placed fragments in the created secret-fragment-visible mosaic image, as a whole, look like the target image instead. Observing such a type of mosaic image, people can see all of the fragments of the secret image, but because the fragments are so tiny in size and so random in position, people cannot figure out what the source image look like unless they have some way to rearrange the pieces back into their original positions, using a secret key from the image owner, as required by the proposed method. As such, the source image may be said to be *secretly* embedded in the resulting mosaic image, though the fragment pieces are all *visible* to an observer of the image. And this is just the reason why we name the resulting image as a *secret-fragment-visible mosaic image*. Examples of such images will be given later in this section.

In the proposed secret-fragment-visible mosaic image creation process, we divide a secret image into a large number of tile images and use them as a mosaic tile

image database. Because the database is formed with secret tile images, the number of tile images is limited; also, the resulting mosaic image includes all of them. Thus, the method for fitting tile images becomes a tough problem to deal with. In this study, we not only develop an algorithm for fitting tile images successfully, but also propose information hiding methods which modify this algorithm in detail to achieve the designs of two information hiding techniques, namely, covert communication via *color* mosaic images and image steganography via *grayscale* mosaic images.

(a) (b)

Figure 1.2 A sliding puzzle. (a) Randomly-placed block pieces of the puzzle. (b) Rearranged block pieces which show the original picture, a self-portrait of Van Gogh.

Each secret-fragment-visible mosaic image is composed of many fragment pieces, and random arrangement of them forms a mosaic pattern, as mentioned previously. Extending this idea, we also use these pieces to compose multiple secret-fragment-visible mosaic images and so implement a secret sharing technique.

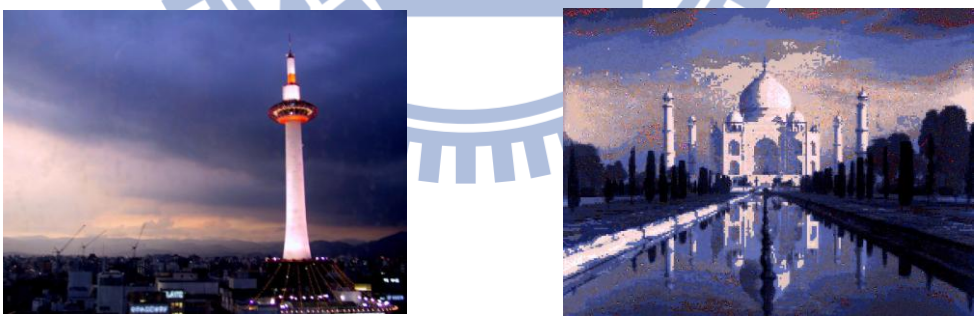
In short, in this study, in addition to designing a new type of mosaic image — secret-fragment-visible mosaic image, we have proposed three applications of such images to information hiding, including image steganography, covert communication, and secret sharing. We will introduce them one by one subsequently.

1.2.2 Examples of Secret-fragment-visible Mosaic Images Created in This Study

Some examples of secret-fragment-visible mosaic images created in this study are shown here. Figures 1.3(a) and 1.4(a) are two secret images used respectively to generate the secret-fragment-visible mosaic images shown in Figures 1.3(b) and 1.4(b).

(a) (b)

Figure 1.3 An example. (a) A secret image. (b) A secret-fragment-visible mosaic image created with (a) as the source image.



(a) (b)

Figure 1.4 Another example. (a) A secret image. (b) A secret-fragment-visible mosaic image created with (a) as the source image.

Observing these images, we see that the resulting mosaic images are quite different from the secret images. Therefore, the proposed creation method may be

regarded as a way of implementing the function of information hiding, though a technique for extracting the secret image from the created image is yet to be designed. More results will be shown in Chapter 3.

1.3 Overview of Proposed Methods

In this study, we propose *two* methods for creating secret-fragment-visible mosaic images. One is for use when the source image is a *full-color* one, and the other is for use when the source image is a *grayscale* one. At the beginning, a scheme for creation of full-color secret-fragment-visible mosaic images is presented. Next, a data hiding method for covert communication is proposed utilizing the characteristics of the proposed secret-fragment-visible mosaic images creation process. Then, a method of image steganography implemented in the process of creating grayscale secret-fragment-visible mosaic images is proposed with the input image transformed from a full-color secret document. Finally, we achieve the goal of secret sharing through the use of multiple secret-fragment-visible mosaic images. Brief descriptions of these methods are given as follows.

1.3.1 Definitions of Terms

Before describing the proposed methods, some definitions of terms used in this study are introduced first as follows.

1. *Secret imag*e: a secret image is one which is chosen as the source image to produce a secret-fragment-visible mosaic image.
2. *Target imag*e: a target image is an image selected from a database according to a secret image, which is the picture stored in the database most similar to

the secret image.

3. *Target image database:* a target image database is a database which store a large number of target images.
4. *Tile image*: a tile image is one of the square pieces resulting from crumbling a secret image.
5. *Target block*: a target block is the place that a tile image of the same size should be fitted into.
6. *Secret-fragment-visible mosaic image*: a secret-fragment-visible mosaic image is obtained by rearranging the fragments of a secret image in a certain way, yielding a visual effect like the target image.
7. *Creation process*: a creation process produces a secret-fragment-visible mosaic image from a secret image.
8. *Recovery process*: a recovery process reconstructs the secret source image from a secret-fragment-visible mosaic image.
9. *Recovery sequence*: a recovery sequence is a sequence which records the corresponding labels of the tile images and the target blocks, based on which the recovery process can be conducted to retrieve the secret image.
10. *Image steganography*: image steganography is a scheme to embed data into images for covert communication, which, though obvious, transmits the secret imperceptibly.
11. *Embedding process*: an embedding process hides secret data into an image.
12. *Extraction process*: an extraction process retrieves secret data from an image.

1.3.2 Brief Description of Proposed Method for

Creation of Secret-fragment-visible Mosaic Images

The main steps of the proposed creation process of secret-fragment-visible mosaic images are described in Figure 1.5. First, we construct a target image database which records the color histogram of the stored images. Second, based on a given secret image, a similarity measure proposed in this study is computed and used to choose a target image from the database. Then, the secret image is divided into tile images, which then are rearranged to fit a target image to create a secret-fragment-visible mosaic image. The detailed creation process will be introduced in Chapter 3.

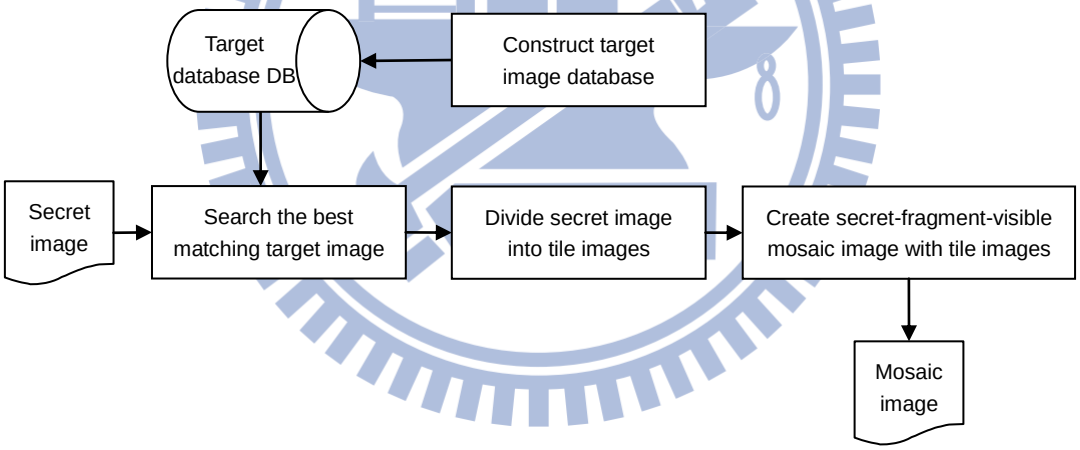


Figure 1.5 Creation process of secret-fragment-visible mosaic image.

1.3.3 Brief Description of Proposed Method for Covert Communication via Secret-fragment-visible Mosaic Images

Based on the method of the above-mentioned creation process, we propose a

covert communication scheme via full-color secret-fragment-visible mosaic images by embedding a source-image recovery sequence and switching the orders of target blocks. A modified version of the secret-fragment-visible mosaic image creation process for use as the secret message embedding process for cover communication is shown in Figure 1.6. First of all, we calculate the 1-dimensional color histograms of a given secret image and a selected target image. Second, the secret messages which are transmitted by users are transformed into a bit string. With the histogram values of a secret image and a target image, we switch the corresponding target blocks of the tile images which are in the same histogram according to each bit to be embedded. After these steps, a secret-fragment-visible mosaic image into which the secret messages are hidden is created by the use of the modified fitting target blocks and a secret key. Finally, a source-image recovery sequence is embedded into the resulting image by a scheme of lossless *least significant bit* (LSB) modification. With the recovery sequence, the secret image can be retrieved quickly and easily. The detailed algorithm is presented in Chapter 4.

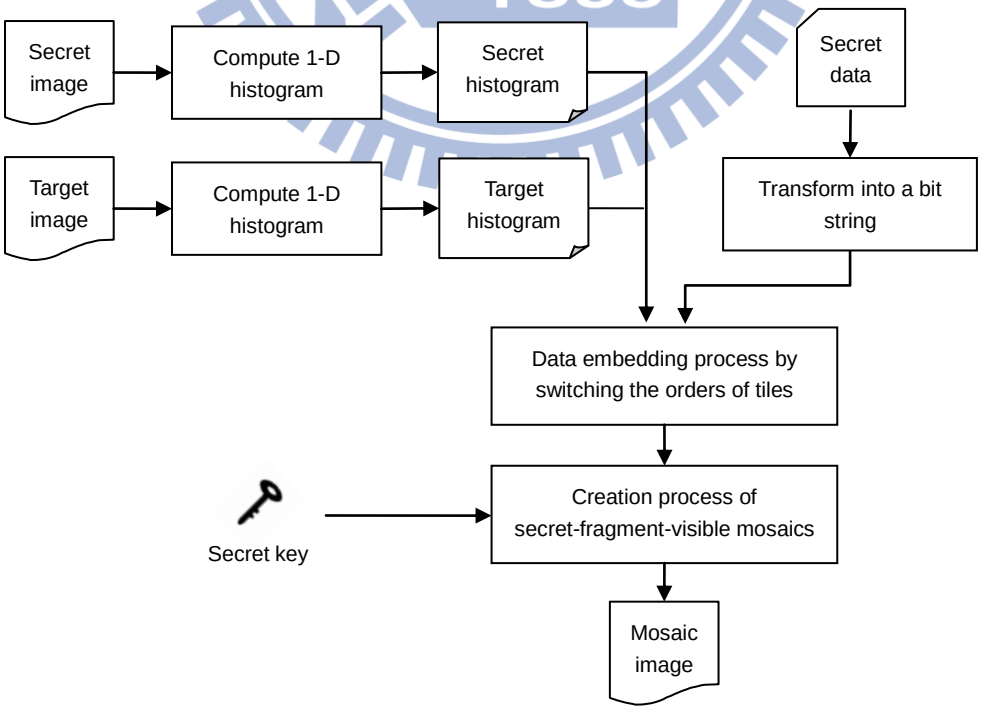


Figure 1.6 Embedding process of data hiding by switching the orders of tiles.

1.3.4 Brief Description of Proposed Method for Image Steganography via Secret-fragment-visible Mosaic Images

The proposed method for image steganography is achieved in the process of creating *grayscale* secret-fragment-visible mosaic images. Grayscale images are still being used frequently in modern days. People can utilize software packages (many available on the Internet) to transform secret documents, such as e-mail, PDF and Microsoft WORD, into grayscale images. We use this type of image to create a secret-fragment-visible mosaic image in order to disguise the transformed secret image as a grayscale art image. With this technique, while we deliver the document disguised as a grayscale mosaic image to other people, hackers will hardly notice that the file is carrying a secret image. And because the transformed image keeps the readability of the original documents by image fragments, a receiver can understand what the document content is after he/she regains the secret image from a secret image recovery process. The detailed algorithms will be given in Chapter 5.

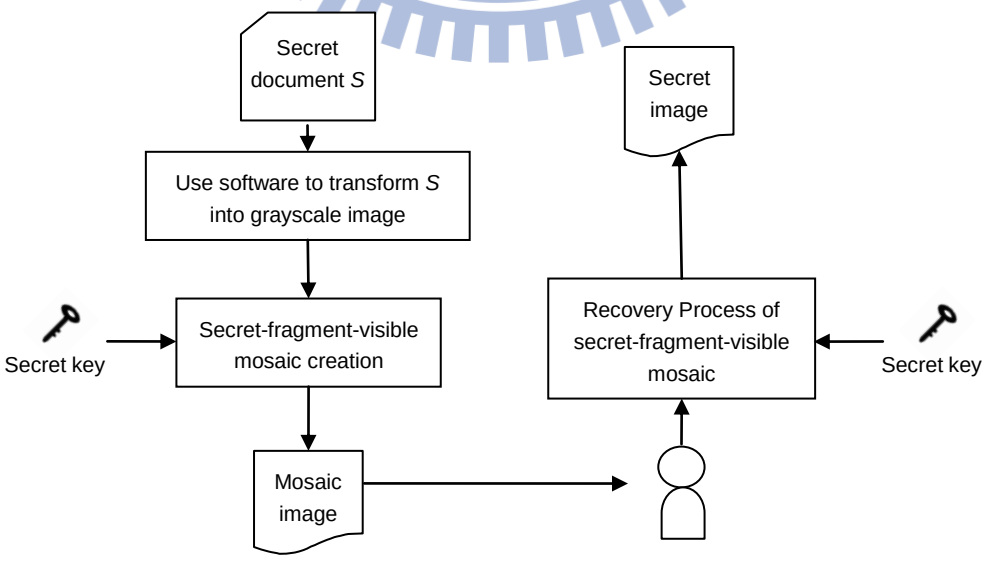


Figure 1.7 Image steganography process by creating grayscale secret-fragment-visible mosaics.

1.3.5 Brief Description of Proposed Method for Secret Sharing via Secret-fragment-visible Mosaic Images

A method for secret sharing is proposed in this study. It is based on partitioning secret tile images into a number, say *n*, of sections and using them separately to create secret-fragment-visible mosaic images. As shown in the process illustrated by Figure 1.8, first, we select *n* target images from a database based on a secret image, and then divide the secret image into tile images of appropriate sizes. After that, we use a key to decide which target image can select tile images first, and then fit the selected tile image to the corresponding target block. The target images will take turns in this way of picking appropriate tile images.

According to the above operations, *n* secret-fragment-visible mosaic images are created from a secret image. Then, the recovery sequence of every mosaic image is divided into $n - 1$ parts and hidden in the resulting *share images*. In this way, the secret image will be retrieved only if all of the share images are collected together. The detailed operations are described in Chapter 6.

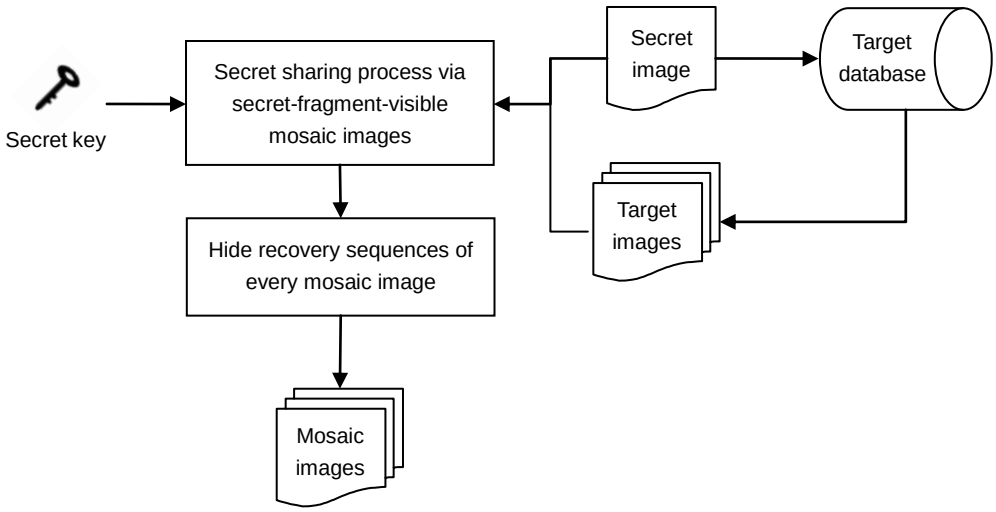


Figure 1.8 Process of secret sharing through secret-fragment-visible mosaic images.

1.4 Contributions

Some major contributions of this study are listed as follows.

1. A method for creation of a new type of mosaic image, called secret-fragment-visible mosaic image, is proposed.
2. A greedy method is proposed, which fits tile images into appropriate target blocks in secret-fragment-visible mosaic images by computing the 1-dimensional color histograms of target images and secret images.
3. An algorithm is proposed, which can be used to decrease the running time of fitting tile images into appropriate target blocks.
4. A remedy method is proposed for reducing the target-block fitting error between a secret image and a selected target image while the image number of a database is not large enough for selecting a sufficiently similar target image.
5. A method is proposed to embed secret messages in secret-fragment-visible mosaic images by switching the corresponding target blocks of tile images.
6. A method for enhancing the security of the embedded secret message is proposed.
7. An algorithm for creating grayscale secret-fragment-visible mosaic images for the aim of image steganography is proposed.
8. An algorithm for modification of the grayscale histogram to decrease the running time of the secret image hiding process for image steganography is proposed.
9. A method for enhancing the security of the secret image hiding process in the proposed method is proposed.
10. A method for secret sharing through the generation of multiple

secret-fragment-visible mosaic images is proposed.

1.5 Thesis Organization

This thesis is organized as follows. In Chapter2, we review the related works of this study. In Chapter 3, the proposed method for creation of secret-fragment-visible mosaic images is described. In Chapter 4, the proposed method for covert communication via full-color secret-fragment-visible mosaic images by embedding a source-image recovery sequence and switching the orders of target blocks is described. Then the proposed method for image steganography though grayscale secret-fragment-visible mosaic images is presented in Chapter 5. In Chapter 6, the proposed method for secret sharing through the use of multiple secret-fragment-visible mosaic images will be introduced. The conclusions of our study and suggestions for future works are included in Chapter 7.

Chapter 2
Review of Related Works

2.1 Previous Studies on Creation and Application of Mosaic Images

Mosaic is an art of creation artworks composed of small pieces of materials which essentially can be of any form or shape. It utilizes a property of human beings' vision that people only can see the average color of a block which is far away from them. Because of this feature, each element in a mosaic image is put at a place which has a similar color to the original image. It is in this way that the mosaic image looks like the original image when seen from a distance. Previously, mosaic images are used to decorate the walls or ceilings of Catholic churches (see Fig. 2.1 for an example). Nowadays, mosaics have become popular and widely used for various purposes of decorations in people's daily life.

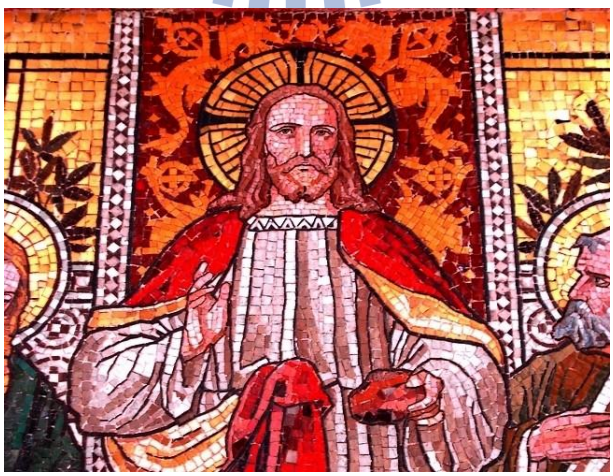


Figure 2.1 Mosaic decorations on the walls of a church. [3]

Many researches on automatic mosaic image creation have been conducted in recent years. How to combine art image creation and the computer technology is a new research topic.

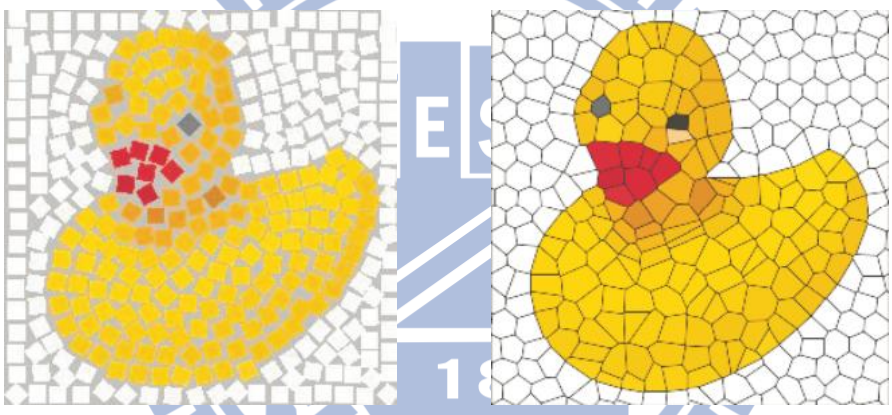
In 1990, Haeberli [4] proposed a method for mosaic image creation. The method uses voronoi diagrams, placing the sites of blocks randomly and filling colors into the blocks based on the content of the original image. It can generate a high-resolution mosaic image from a given image, like Figure 2.2(a). Hausner [5] presented a method to create tile mosaic images by using centroidal voronoi diagrams. He made good use of it because the covered space is fair. Two example images are shown in Figures 2.2(b) and 2.3(a). Extending Haeberli's idea, Dobashi et al. [6] improved voronoi diagrams to create tile mosaic images, like Figure 2.3(b). The method of Dobashi consists of two processes. One is to generate voronoi diagrams and optimize them by reducing the matching error value between a source image and a resulting image. The other process allows a user to add various effects to the mosaic image, such as simulation of stained glasses.

(a) (b)

Figure 2.2 Munch's "The Scream". (a) Using Haeberli's [4] method. (b) Using Hausner's [5] method.

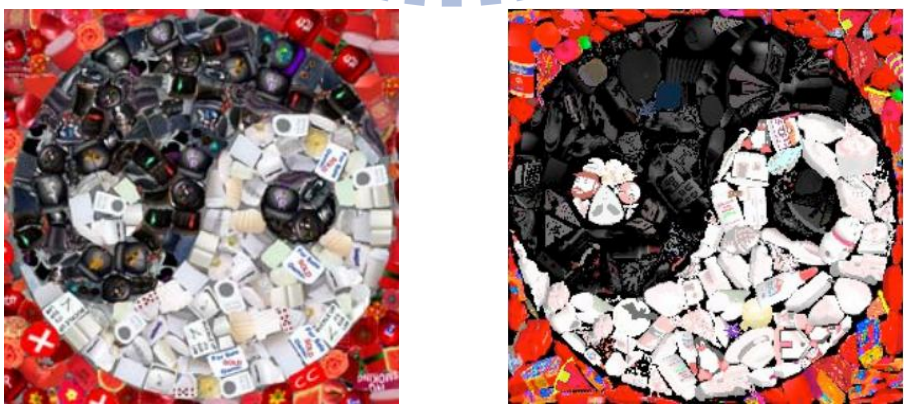
A third type of mosaic image is composed of tile images which have arbitrary

shapes. As shown by Figures 2.4(a) and 2.4(b), the final image is filled with small elements of different shapes. Kim and Pellacini [7] proposed a creation process for generating this type of mosaic image, called *jigsaw image mosaic*. It is composed of many arbitrary shapes of tiles which are selected from a database. However, the tiles may be distorted in the proposed algorithm. Extending the concept of Kim's method, Blasi et al. [8] presented a new mosaic image called *puzzle image mosaic* which is similar to Kim's results. The creation times of puzzle image mosaics are less than those of jigsaw image mosaics, and better effects with no distortion of tile images were obtained.



(a) (b)

Figure 2.3 Mosaic Images. (a) Using Hausner's [5] method. (b) Using Dobashi's [6] method.



(a) (b)

Figure 2.4 Mosaic images. (a) Jigsaw image mosaic [7]. (b) Puzzle image mosaic [8].

2.2 Previous Studies on Information Hiding Techniques

Information hiding is a technique which changes the insensitive parts of a file format in order to embed some extra messages in it without consciousness of other people. The most well-known method is least significant bit (LSB) modification. Many studies extended this idea of LSB modification method and utilized it in many applications. Some of them embed secret images into cover images in order to implement covert communication and invisible watermarking. For example, Wu and Tsai [9] presented a data hiding method according to a human vision model. They hid secret data in the sharp image area based on the characteristics of human vision. This method can embed secret images losslessly and create stego-images with low degradation. However, cover images with traditional LSB modifications may be changed and cannot be reversed to the original ones. Based on this drawback, many researches were conducted to implement reversible LSB modification techniques.

Fridrich et al. [10] proposed a lossless LSB modification method with flipping functions and pixel grouping. They divided the pixels into three groups and applied flipping functions to each group. Celik et al. [11] presented a method, called *lossless generalized LSB data embedding*. The proposed algorithm modifies the lowest levels, instead of bit planes, of raw pixel values, and used the results as features. The recovery of the original image is achieved by compressing, transmitting, and recovering these features. Tian [12] proposed a reversible data embedding method by using a difference expansion scheme. The method uses some simple equations to modify the values of two pixels. Because the new values are generated from the difference between two manipulated pixels, the original pixel values can be recovered easily. Alattar [13] proposed a lossless data hiding method using the difference

expansion of a generalized integer transform. This method extends Tian's algorithm utilizing difference expansion of vectors, instead of pairs, to increase the hiding ability of the method.

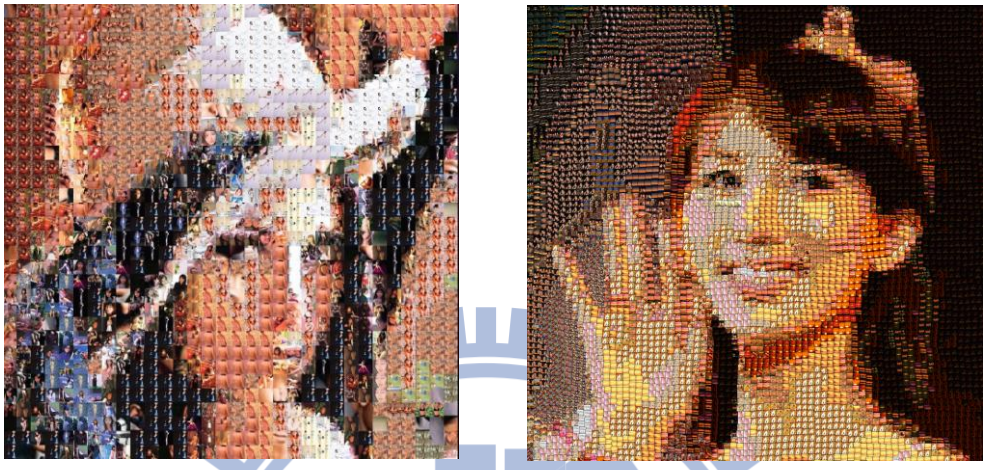
The above-mentioned methods all have the drawback of yielding low data embedding capacities. Coltuc and Chassery [14] presented a high-capacity data embedding scheme without appealing to any additional data compression stage. This scheme is based on a reversible contrast mapping, which is a simple integer transform defined on pairs of pixels. They marked pairs of pixels as three groups, and applied different operations on them. In this study, we utilize this scheme to hide a source-image recovery sequence into a secret-fragment-visible mosaic image. In order to embed other secret messages into the mosaic image generated in this study, the method designed for hiding the recovery sequence must be reversible and has a high data embedding capacity.

2.3 Previous Studies on Information Hiding Techniques in Art Images

The combination of art image creation and information hiding techniques is a new concept of computer technology. This technique utilizes the characteristics of the art image creation process to embed extra information in the generated images. With this disguise, hackers will tend to get unaware of the data embedded in such images, and secret data can so be kept or transmitted safely and covertly.

Lin and Tsai [15] proposed methods for embedding secret data in image mosaics by adjusting regions of boundaries and altering pixel values of the hue component in the HIS color model. A result generated by the method is shown in Figure 2.5(a). Wang and Tsai [16] also presented a data hiding method for image mosaics. By

utilizing the overlapping space of component images, the scheme can embed secret messages in image mosaics without arousing notice from an observer. An example of the resulting images is shown in Figure 2.5(b).



(a) (b)

Figure 2.5 Image mosaics. (a) An image mosaic created with Lin and Tsai's method [15]. (b) An image mosaic created with Wang and Tsai's method [16].

In addition to generating image mosaics, several studies of combining art image creation and information hiding have been conducted, yielding other types of art images. Hung and Tsai [1] proposed information hiding methods through the use of stained glass images and tile mosaic images. By modifying the tree structure used in the creation process, secret data can be hidden in computer-generated stained glass images. Moreover, they utilized the rotation angles of the tiles in tile mosaic images to design data hiding techniques. Hsu and Tsai [17] presented three new types of art images and used the characteristics of their creation processes to hide secret messages in the generated art images. First, in the digital puzzle image generated by them, the angles of puzzles are used for data hiding. Second, they proposed a method for generating a kind of so-called *circular-dotted image*, as shown in Figure 2.6(a). By changing the order of circular-dot overlappings, a method of information hiding was

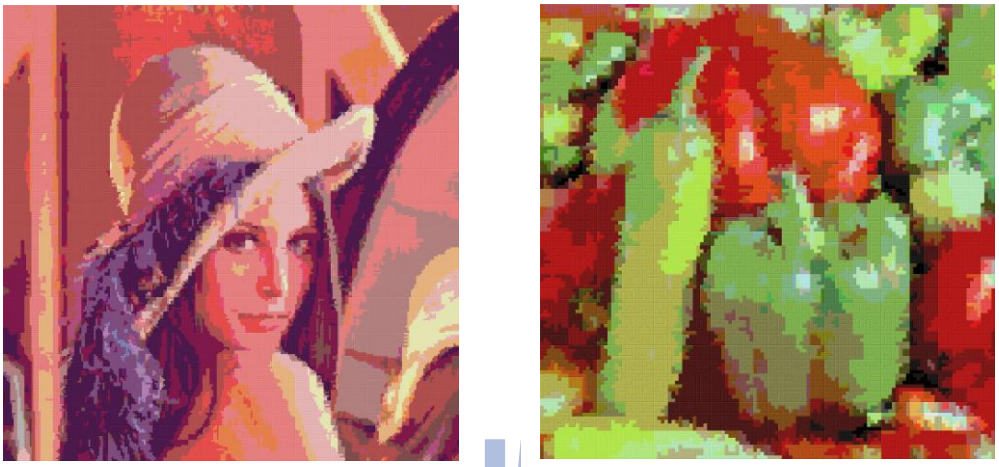
implemented. And the last data hiding method proposed by them is the use of the pallet colors in pointillistic images for data hiding. An image yielded by their method is shown in Figure 2.6(b). Chang and Tsai [18] proposed a new type of art image, called tetromino-based mosaic, which is composed of tetrominoes appearing in a video game. By geometric shape composition, tetrominoes can be combined to one another to form blocks which in turn can be used to fill a plane with a limited shape (rectangles mostly). This is the reason why tetrominoes can be used to create mosaic images. Data hiding is made possible by distinct combinations and color shifting of the tetromino elements. Examples of the resulting images are shown in Figures 2.7(a) and 2.7(b).

(a) (b)

Figure 2.6 Art images created by Hsu and Tsai [17]. (a) A circular-dotted image. (b) A pointillistic image.

In this study, we also propose new methods which combine art image creation and information hiding techniques to implement covert communication, image steganography, and secret sharing. By using the characteristics of the secret-fragment-visible mosaic image creation process, the images can be transmitted

with the embedded data arousing no notice from observers.



(a) (b)

Figure 2.7 Tetromino-based mosaic images [18]. (a) Lena. (b) Peppers.

Chapter 3
Creation of Secret-fragment-visible Mosaic Images

3.1 Idea of Proposed Method

A sliding puzzle is composed of all block pieces but one of a divided image. The randomly-placed block pieces compose a *meaningless* picture. Players can slide the blocks in a certain way by observing the color or texture of them, and finish the game with a *meaningful* picture. The idea of the proposed art image creation method is inspired by this concept of *puzzle piece sliding*, as mentioned previously, to create the new type of mosaic image, the *secret-fragment-visible mosaic image*, as proposed in this study.

In the creation of a secret-fragment-visible mosaic image, a secret image is divided into fragment pieces. The pieces of the secret image are all visible in the mosaic image, but the size of them is tiny and the rearranged positions are random. Therefore, people cannot extract the secret data from the mosaic image unless they have the knowledge to rearrange the pieces back into their original positions, using the right secret key from the owner as required by the proposed method.

The proposed mosaic image creation process is composed of two major procedures. The first is the construction of an image database which can be used later to select a sufficiently similar target image for a given secret image. The quality of a constructed secret-fragment-visible mosaic image is related to the similarity between the secret image and the target image. So we try to select a target image from a

database based on the contents of a given secret image by computing a similarity measure; the selected target image should be as similar to the secret image as possible.

Another procedure is the creation of a secret-fragment-visible mosaic image, which uses the secret image and the selected target image as input. In this procedure, we divide a secret image into fragment pieces, i.e., *tile images* as mentioned before, and use these tile images to create the mosaic image. The number of tile images for use in mosaic image creation is *limited* by the size of the secret image and that of the tile images. Note that this is not the case in traditional mosaic image generation where available tile images to fit into the target image are *unlimited in number*. In order to solve this problem of fitting a limited number of tile images into a target image, we propose a greedy algorithm to find an appropriate answer.

In addition, we present a method for remedying a database which is not large enough for selecting a sufficiently similar target image. The method basically enlarges the size of the selected target image so that the tile images of the secret image can be fitted into the target image at more freely-chosen positions, resulting in a mosaic image more similar to the secret image.

The detailed algorithms of the above-mentioned processes are presented in the following sections.

3.2 Proposed Secret-fragment-visible Mosaic Image Creation Process

3.2.1 Database construction

First of all, we have to construct a database which keeps target images. The

database plays an important role in the secret-fragment-visible mosaic image creation process. If a target image is dissimilar to a secret image, the created image will be distinct from the target one. In order to generate a good result, the database so should be as large as possible.

To search a target image from a database with the highest similarity to the secret image is a problem of *content-based image retrieval*. In general, the content of an image may be described by features like shape, texture, and color. Because only the color distributions of a secret image and a target image will affect the overall visual appearance of the resulting mosaic image, we may focus first on extracting color distributions from images by techniques developed for content-based image retrieval.

The simplest technique for extracting the color distribution of an image by content-based image retrieval is 1-D color histogram transformation [19]. This technique transforms three color channel values (such as *R*, *G*, and *B* or *H*, *S*, and *V*) into a single channel value. More details are described in the following.

First, each color channel (usually with a range of 0~255, i.e., with 256 levels) is re-quantized into less levels, yielding an new image I' with a lower resolution in color, which we specify as (r, g, b). Let N_r, N_g, and N_b denote the numbers of levels for the new colors r, g, and b respectively. Then, for each pixel P' in I' with new color (r', g', b'), we compute the following 1-D function value f:

$$f(r', g', b') = r' + N_r \times g' + N_r \times N_g \times b' \tag{3.1}$$

and so generate conceptually a 1D image I'' with new "1-D color" values specified by f above. Then, this new image then can be used for image content analysis and similar image search and retrieval. Note that the sizes of I' and I'' are both the same as that of the original image I.

However, according to our experimental experience using this 1-D color function

f, it is found inappropriate for our study here which emphasizes human's visual feeling of image similarity. Therefore, we propose a new function *h* as follows:

$$h(r', g', b') = b' + N_b \times r' + N_b \times N_r \times g' \quad (3.2)$$

where the numbers of levels, N_r, N_g and N_b, are *all set to 8*. Differently from the case in (3.1), we set in (3.2) the largest weight $N_b \times N_r$ to the green channel value g' and the smallest weight 1 to the blue channel value b'. The reason is that the eyes of human beings are the most sensitive to the green color, and the least sensitive to the blue color. In addition, with all of N_r, N_g, and N_b set to 8 in (3.2), an advantage of speeding up the later process of mosaic image creation can be obtained according to our experiments. In the sequel, we will say that the new color feature function *h* we propose above defines a 1-D *h-colorscale*.

Furthermore, to measure the similarity between a *tile image* in the secret image and a *target block* in an image in a database for use in tile image fitting in generating a mosaic image, we propose a new feature, called *h*-feature, for each block image *C* (either a tile image or a target block), denoted as h_C, which is computed by the following steps:

1. compute the average of the color values (*R*, *G*, *B*) of all the pixels in *C* as (R_C, G_C, B_C);
2. re-quantize (R_C, G_C, B_C) into (r_C', g_C', b_C') using the new N_r, N_g, and N_b color levels; and
3. calculate the *h*-feature h_C for *C* by Eq. (3.2) above, resulting in the following equation:

$$h_C(r_C', g_C', b_C') = b_C' + N_b \times r_C' + N_b \times N_r \times g_C'. \quad (3.3)$$

With *Nr*, N_g, and N_b all equals 8, the range of the computed values of the

h-feature f_C above may be figured out to be from 0 to 584. The proposed algorithm for constructing a database of *candidate images* (from which a target image is to be selected for each given secret image) for this study is described in the following.

Algorithm 3.1: ***candidate image database construction.***

Input: a set *S* of images, a pre-selected tile image size Z_t, and a pre-selected candidate image size Z_c.

Output: a database *DB* of candidate images with size Z_c and their *h-colorscale* histograms.

Steps:

Step 1. For each input image *I*, perform the following steps.

1.1 Resize and crop *I* to yield an image *D* of size Z_c.

1.2 Divide *D* into blocks of size Z_t.

1.3 For each block *C* of *D*, calculate and round off the *h*-feature value h_C described by Eq. (3.3).

1.4 Generate a histogram *H* of the values of the *h*-features of all the blocks in *D*.

1.5 Save *H* together with *D* into the desired database *DB*.

Step 2. If the input images are not exhausted yet, then go to Step 1; otherwise, exit.

3.2.2 Similarity measure computation and target image selection

Before we generate a mosaic image, we have to choose as the target image a similar candidate image from the database based on the given secret image content. For this, we can use the 1-D *h*-colorscale histogram of each candidate image in the database to compute a similarity measure between the secret image and the candidate

image. More specifically, by summing up the differences between the 1-D histograms of the secret image and that of the candidate image, we can get an *error e* (computed by Eq. (3.4) described later in the following). The smaller the value *e* is, the more similar the candidate image is to the secret image. After calculating the errors of all the candidate images in the database, we can select the candidate image with the smallest error as the desired target image for use in later mosaic image generation. The detailed algorithm is given as follows.

Algorithm 3.2: ***selection of the most similar candidate image to be the target image.***

Input: a secret image *S*, a database *DB* of candidate images, and the sizes Z_t and Z_c of tile images and candidate images, respectively, mentioned in Algorithm 3.1.

Output: a target image *T* selected from *DB* with the largest content similarity to *S*.

Steps:

Step 1. Resize *S* to yield an image *S′* of size Z_c to become of the same size as the candidate images in *DB*.

Step 2. Divide *S′* into blocks of size Z_t, and perform the following steps.

2.1 For each block *C* of *S′* generated in Step 1, calculate and round off for it the *h*-feature value h_C described by Eq. (3.3).

2.2 Generate a 1-D *h*-colorscale histogram $H_{S'}$ for *S′* from the values of the *h*-features of all the blocks in *S′*.

Step 3. For each candidate image *D* with 1-D *h*-colorscale histogram H_D stored in *DB*, perform the following steps.

3.1 Compute an error *e* as the similarity measure between $H_{S'}$ and H_D by the following equation:

$$e = \sum_{m=0}^{584} \left| H_{S'}(m) - H_D(m) \right|; \tag{3.4}$$

where *m* stands for the value of the *h*-feature mentioned above.

3.2 Record the error *e*.

Step 4. If the images in *DB* are not exhausted, then go to Step 3; otherwise, continue.

Step 5. Select the image in *DB* which has the minimum error value and take it as the desired target image *T*.

3.2.3 Algorithm for secret-fragment-visible mosaic image creation

In this section, after discussing some problems which are encountered in the creation process of secret-fragment-visible mosaic images, we will present an algorithm to implement the process.

A. Problems of creating secret-fragment-visible mosaic images

We face two major problems in the secret-fragment-visible mosaic image creation process. One is about finding an optimal solution for fitting tile images in appropriate target blocks. Another is dealing with the situation of a database which is not large enough for selecting a sufficiently similar candidate image as a target image.

About the first problem, we can reduce it to a single-source shortest path problem. The shortest path problem is one of finding a path in a graph with the smallest sum of between-vertices edge weights. The state of fitting tile images represents the vertices of the graph. In the fitting process, we select a tile image for a target block, once a block. Therefore, the edge means the label of the tile image that we choose at the time. Then, the weight of an edge is the Euclidean distance between the selected tile image and the filled target block (defined later). Accordingly, we can build a tree structure for this problem, as shown by Figure 3.1.

In order to find an optimal solution of the single-source shortest path problem, we utilize Dijkstra's algorithm. In this algorithm, the running time of getting an

optimal answer is $O\left(|V|^2\right)$, where V stands for the number of vertices. According to Figure 3.1, the number of vertices in this problem is as follows:

$$\sum_{n=1}^{N-1}\left[\frac{(N-1)!}{n!}\right],$$

where N is the number of target blocks which is larger than 40,000 for the images used in this study. Obviously, the computation cost for getting an optimal solution for such a large N is too high. In this case, we have to find other possible solutions for fitting tile images in order to create a secret-fragment-visible mosaic image.

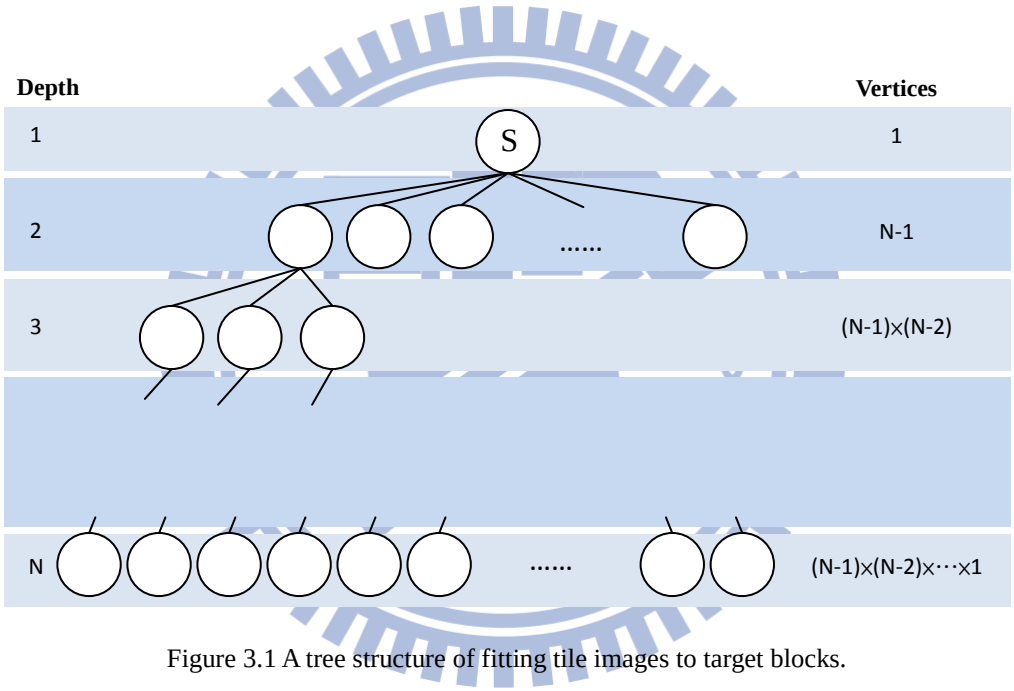


Figure 3.1 A tree structure of fitting tile images to target blocks.

The second problem is about how to select a sufficiently similar target image from a database which is not large enough. While we use such a database in the secret-fragment-visible mosaic creation process, the process will very likely to choose a dissimilar target image. As a result, the resulting image will look unlike the target one. In this case, we have to design a remedy method to deal with such a problem of use of a small-sized database.

B. Possible solutions for fitting tile images

One possible solution for fitting tile images is the *greedy algorithm*. We try to calculate the Euclidean distance between a tile image and a target block as a selection function for a greedy algorithm. However, as shown by Figure 3.3(a), the resulting image of using such a greedy algorithm for fitting tile images is *incomplete* with its lower part being filled with fragment pieces with inappropriate colors. This phenomenon comes from the reason that the number of tile images obtained from the secret image, Figure 3.2(a), is limited by its own size, so that the available tile images for choice to fit the target blocks in Figure 3.2(b) become less and less near the end of the fitting process. As a result, the differences in terms of the Euclidean distance between the later-fitted tile images and the target blocks become bigger and bigger than the earlier-fitted ones, resulting in the poorly-fitted bottom part of Figure 3.3(a).

To get a possible solution to this problem, we try to vary the select function for use in the greedy algorithm. After many trials, we find that the effect on the created mosaic images becomes better if the previously-proposed *h*-feature, instead of the Euclidean distance, is used for defining the selection function for the greedy algorithm. This feature takes the global color distribution of an image into consideration. That makes the content of a created mosaic image using the proposed creation process resemble a target image, as shown by Figure 3.3(b)

C. Remedy method for creation process

By using the greedy algorithm with the selection function defined in terms of the *h*-feature, we can generate a secret-fragment-visible mosaic image successfully. Yet we still have the small-sized database problem to deal with. While a database is not large enough, the mosaic image creation process will select from the database a candidate image dissimilar to the secret image. As shown by Figure 3.5(a) which is a created mosaic image, which is created with Figures 3.4(a) and 3.4(b) as the secret

and the selected target image, the major color of the created mosaic image is quite different from the selected target one.

(a) (b)

Figure 3.2 Input images. (a) A secret image. (b) A target image.

(a) (b)

Figure 3.3 Created mosaic images generated by a greedy algorithm. (a) Image created using Euclidean distance used to define a selection function for a greedy algorithm. (b) Image created using the *h*-feature to define the select function for a greedy algorithm.

To solve this problem, during the candidate image selection process, after we

compute an error between a secret image and a candidate image to measure the similarity, if the error is large, the selected target image is inappropriate for the creation process. In this case, we propose to *enlarge* the size of the target image. The reason is that if the size of a target image is larger than that of the secret image, the number of target blocks, or equivalently, the number of positions to fit the tile image, may be more freely chosen.

Furthermore, to deal with the *surplus* target blocks coming from enlargement of the target image, we fill each of them with the most similar tile image in the sense of the *h*-feature or with the average color of itself. More specifically, if the difference between the *h*-feature of the most similar tile image and that of the target block is larger than a certain threshold, then the target block is filled in the average color instead of the selected tile image. In this way, the resulting mosaic image is better than before, as shown by Figure 3.5(b).

D. Algorithm for secret-fragment-visible mosaic image creation

According to above-mentioned related processes, an algorithm for the proposed secret-fragment-visible mosaic image creation is described in the following. Basically, in the process of tile image fitting, we try to sort the *h*-feature values of tile images and target blocks, and then get two block-label sequences from the sorted feature values of them. By one-by-one mapping these two block-label sequences, each tile image can be fit into a most similar target block achieving the greedy search goal and decreasing the running time to $O(n \log n)$. The detailed algorithm is given as follows.

Algorithm 3.3: ***secret-fragment-visible mosaic image creation.***

Input: a secret image *S*, a database *DB*, and a selected size Z_t of a tile image.

Output: A secret-fragment-visible mosaic image R.

Steps:

Stage 1 --- embedding secret image fragments into a selected target image.

Step 1. Crop S to yield an image S' which is divisible by size Z_t.

Step 2. Perform following steps to select a target image T.

2.1 Select a candidate image by Algorithm 3.2 as T.

2.2 If the error e computed in Step 3.1 of Algorithm 3.2 is larger than a pre-selected threshold T_h, then enlarge the size of T $\lceil e/T_h \rceil$ times.

Step 3. Perform the following steps to obtain a *block-label sequence* of each of S' and T.

3.1 Divide S' into a sequence $Q_{S'}$ of blocks S_1', S_2', …, S_n' as tile images based on size Z_t, where each block S_i' is said to have the *label i*.

3.2 Divide T into a sequence Q_T of blocks T_1, T_2, …, T_n as target blocks based on size Z_t, where the label of T_i is similarly defined.

3.3 For each tile image of S' and each target block of T, calculate the h-feature values of them based on Eq. (3.3).

3.4 Sort the h-feature values of all S_i' and all T_i, and re-order accordingly the blocks in $Q_{S'}$ and Q_T, respectively, to get two *re-ordered block sequences* $Q_{S'}'$ and Q_T'.

3.5 Get the new label sequences L_1 and L_2 from the re-ordered block sequences $Q_{S'}'$ and Q_T', and call them *h-sorted label sequences* of the resized secret image S' and the selected target image T.

Step 4. Fit the tile images to the target blocks based on the one-to-one mappings from the ordered labels of L_1 to those of L_2, thus completing the embedding of all the tile images in S' into all the target blocks of T.

Stage 2 --- dealing with unfilled target blocks.

Step 5. Perform the following steps to fill each of the remaining unfilled target blocks, B, if there is any.

12.1 Compute the difference e' between the h-feature h_B of B and the h-feature h_A of each of the tile images, A, by the following equation:

$$e' = |h_B - h_A|. \tag{3.5}$$

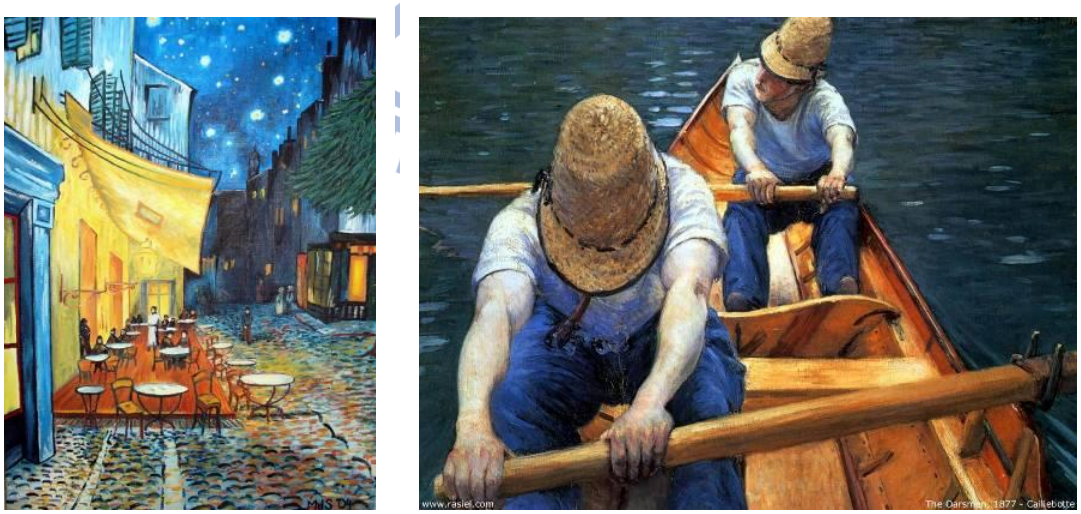
12.2 Pick out the tile image A_o with the smallest error e' and compare e' with another pre-selected threshold T_h' in the following way:

A. if $e' < T_h'$, then fill the tile image A_o into the target block B;

B. if $e' \geq T_h'$, fill the average R, G, and B values of all the pixels in B into B.

Stage 3 --- generating the desired mosaic image.

Step 6. Generate the desired output image R by composing all the tile images fitted at their respective positions as an image.



(a) (b)

Figure 3.4 Example images. (a) A secret image. (b) A target image.

(a) (b)

Figure 3.5 Resulting mosaic images created with Figures 3.4(a) as secret image and 3.4(b) as target image. (a) Mosaic image created without the remedy method. (b) Mosaic image created with the proposed remedy method.

3.3 Experimental Results

Some created mosaic images using the above-proposed algorithms are given in Figures 3.6 through 3.9. The number of candidate target images which are stored in the database is 841. The size of the database is large enough because the remedy method is rarely used in the creation process in our experiments. By calculating the Euclidean distances between the (*r*, *g*, *b*) colors of all the blocks of a target image and those of all the blocks of a corresponding secret-fragment-visible mosaic image, we can get the average error at the block level in the resulting image (as the sum of all the Euclidean distances divided by the number of blocks), as shown by Table 3.1. We use this error measure instead of the *peak signal-to-noise ratio* (PSNR) to examine the similarity between a secret image and a selected target image. Because the visual effects of the mosaic image are based on a property of human vision that people only can see the average color of a block when it is far away from them, it is inappropriate to apply a pixel-level similarity measure for this kind of image. Therefore, we use a block-level similarity as mentioned above. The smaller the value is, the more similar

the secret image to the target image is. Accordingly, from Table 3.1 we see that Figure 3.8(c) is the most-similar mosaic image created by the proposed method among the four of Figures3.6 through 3.9.

(a)

(b)

(c)

Figure 3.6 Experimental images. (a) A secret image. (b) A target image. (c) A secret-fragment-visible mosaic image created with (a) as a source image.

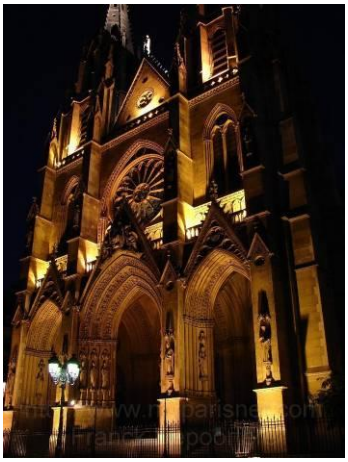
(a)

(b)

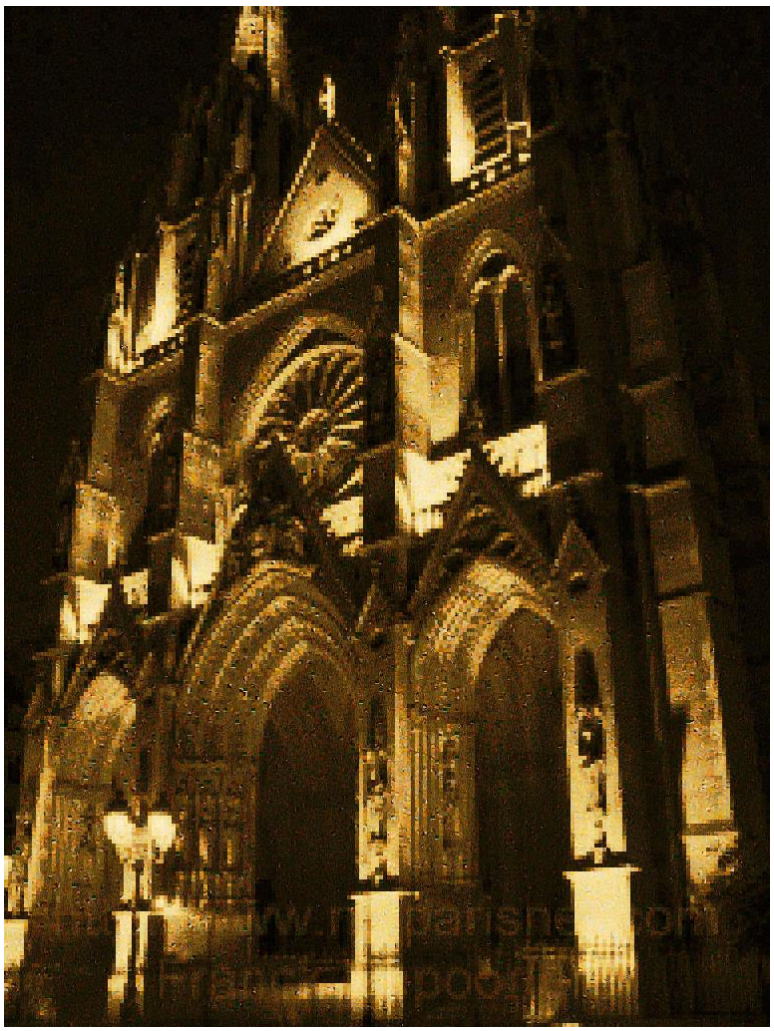
(c)

Figure 3.7 Experimental images. (a) A secret image. (b) A target image. (c) A secret-fragment-visible mosaic image created with (a) as a source image.

(a)



(b)



(c)

Figure 3.8 Experimental images. (a) A secret image. (b) A target image. (c) A secret-fragment-visible mosaic image created with (a) as a source image.

(a)

(b)

(c)

Figure 3.9 Experimental images. (a) A secret image. (b) A target image. (c) A secret-fragment-visible mosaic image created with (a) as a source image.

Table 3.1 The Euclidean distance at block level between target images and created mosaic images.

Figure Number	Euclidean distance at block level
3.6	66.9
3.7	80.8
3.8	39.4
3.9	40

3.4 Summary

A new type of mosaic image, called secret-fragment-visible mosaic image, has been proposed. To accomplish this, first we extended a technique of content-based image retrieval by proposing a new 1-D *h*-colorscale described by Eq. (3.2) to represent the color distribution of an image which fits better according to human visual feeling of an image. With this *h*-colorscale, we have proposed next a new *h*-feature for image similarity measure computation. In order to select the most similar candidate image from a database, we make good use of the *h*-feature value to calculate the error between a secret image and each candidate image in the database as their similarity measure. Besides, a remedy method has been proposed for a database which is not large enough to select a sufficiently similar candidate image. In addition to being used in the image database construction and the similarity measure computation, the *h*-feature value is also used in the process of fitting similar tile images to target blocks. With a greedy method algorithm which uses a selection function computed in terms of the *h*-feature value, we can get a good solution to the problem of fitting tile images.

Chapter 4
Covert Communication via Secret-fragment-visible Mosaic Images

4.1 Idea of Proposed Method

The proposed method of using secret-fragment-visible mosaic images for covert communication is described in this chapter. We describe the basic idea of the proposed method in this section first.

In the proposed secret-fragment-visible mosaic image creation process as described in the last chapter, we use the newly-proposed *h*-feature values of image blocks as a select function for a greedy algorithm to fit tile images into appropriate target blocks. The *h*-feature value h_C for each image block C (a tile image or a target block) is computed according to (3.3), which is repeated as follows:

$$h_C(r_C', g_C', b_C') = b_C' + N_b \times r_C' + N_b \times N_r \times g_C', \tag{4.1}$$

where r_C', g_C', and b_C' are the average R, G, and B values of all the pixels in C. Because the values of r_C', g_C', and b_C' are float-point numbers, after calculating the value h_C by Eq. (4.1) which is also a float-point number, we round it off to be an integer. By this, the blocks which have similar *h*-feature values can be classified to the same group when constructing the *h*-colorscale histogram.

More specifically, with Figure 4.1 shown as an example, tile images with the same *h*-feature values appear to have similar colors. In the mosaic image creation

process as described in the last chapter, each tile image is fitted into a corresponding target block according to the one-to-one mappings established between the two *h*-sorted label sequences of the secret image and the selected target image. The main idea of secret embedding in the proposed covert communication is to *switch the orders of the target blocks in the h-sorted label sequence* of the target image during the mosaic image creation process to embed message bits, thus hiding the secret message into a secret-fragment-visible mosaic image imperceptibly without arousing notice from observers. More details will be described in the subsequent sections.

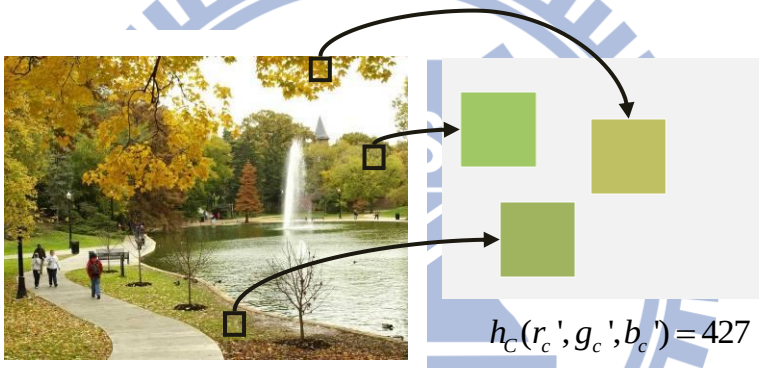


Figure 4.1 Different colors of tile images which have the same *h*-feature value.

4.2 Proposed Secret Message Hiding Method via Secret-fragment-visible Mosaic images

4.2.1 Modified secret-fragment-visible mosaic image creation process for secret message embedding

In the creation process of secret-fragment-visible mosaic images, we get two *h*-sorted label sequences of *h*-feature values of the tile images and the target blocks of

the secret image and the target image, respectively, by Eq. (4.1). Since each of the labels is unique, we may utilize each pair of target block labels and switch the order of them to embed a secret message bit before mapping the two *h*-sorted label sequences for mosaic image creation.

Then, after label switching, if a leading label is smaller than the following one in the target block label sequence, it means that the embedded bit is "0." On the other hand, if we want to hide a bit "1," we make the leading label to be larger than the following one.

As shown by Figure 4.2, because the tile images which correspond to the target blocks with switched labels have the similar average color, after the secret message is embedded, no perceptible difference will arise in the resulting mosaic image. Accordingly, we can modify the algorithm of the secret-fragment-visible mosaic image creation process described in the last chapter for embedding secret messages.

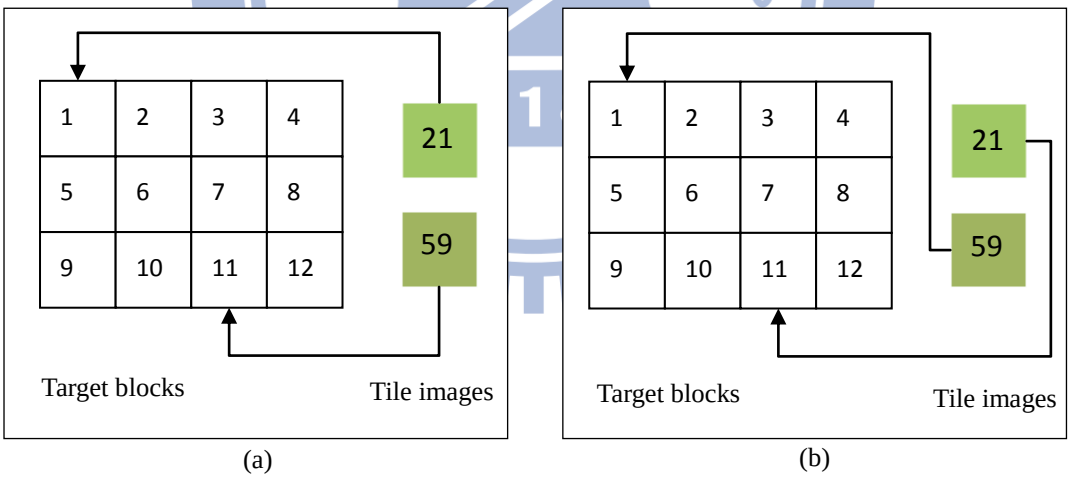


Figure 4.2 Exchange of the corresponding target blocks of tile images. (a) The original corresponding target blocks of tile images. (b) After switching the corresponding target blocks of tile images.

In the proposed covert communication method, mappings of the labels of the tile images to those of their corresponding target blocks will be recorded in a *recovery sequence* L_R for use later in data recovery. In the process of embedding L_R, we hide

the labels of L_R into the tile images which are randomly selected by a secret key. As shown by Figure 4.3, the recovery sequence L_R is supposed to be composed of corresponding labels of tile images and target blocks. However, to reduce the volume of the embedded bits, we record only the labels of L_2 of the target blocks. This can be done by a re-ordering of the labels in L_2, which is based on the label number of their corresponding labels in L_1 — if a label's corresponding label number in L_1 is the smallest, it will be embedded first.

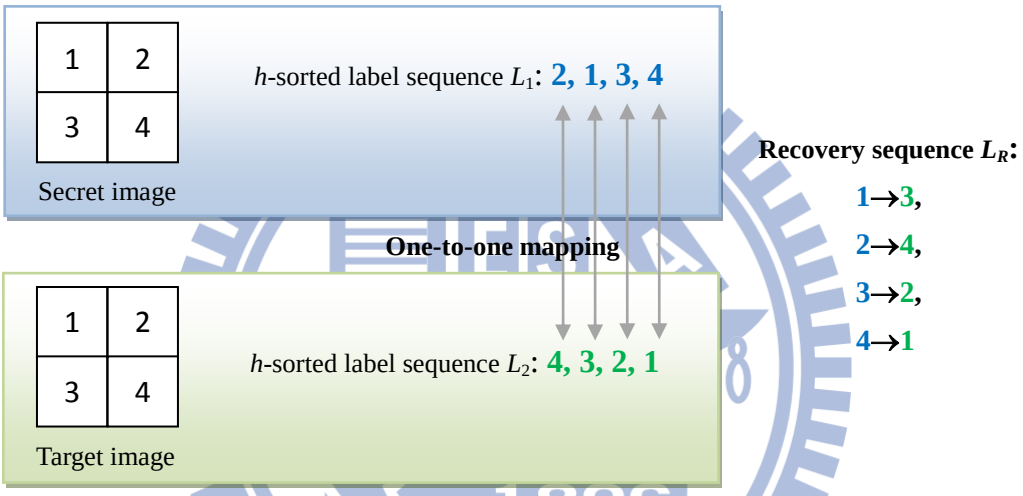


Figure 4.3 An illustration of generation of a recovery sequence L_R.

With a transformation of the recovery sequence L_R into a bit string, we hide it by lossless LSB modification into the tile images of the secret image S fitted in the resulting mosaic image. The number N_P of the bits so embedded, according to the previous discussions (in the last paragraph), may be computed according to the following equations:

$$N_T = \frac{W_S \times H_S}{Z_t}; \tag{4.2}$$

$$L_B = \lfloor \log_2 (N_T) \rfloor + 1; \tag{4.3}$$

$$N_P = L_B \times N_T; \tag{4.4}$$

where N_T is the number of tile images; W_S and H_S represent the width and height of the secret image S, respectively; Z_t is the size of a tile image; and L_B is the length of the binary number of N_T. That is, based on Equations (4.2) through (4.4), we can get *the number N_P of bits to be embedded for embedding L_R*.

The algorithm of embedding a secret message into a secret-fragment-visible mosaic image is given in detail as follows.

Algorithm 4.1: ***embedding a message into a secret-fragment-visible mosaic image.***

Input: a secret image S, a secret key K, the size Z_t of tile images, a candidate image database DB, and a secret message M.

Output: a secret-fragment-visible mosaic image R into which M is embedded.

Steps:

Stage 1 --- embedding secret image fragments into a selected target image.

Step 1. Crop S to yield an image S' with a size *divisible* by Z_t.

Step 2. Perform the following steps to select for S' a target image T with histogram H from the database DB.

2.3 Select a target image T by Algorithm 3.2.

2.4 If the error e computed in Step 3.1 of Algorithm 3.2 is larger than a pre-selected threshold T_h, then enlarge the size of T for $\lceil e/T_h \rceil$ times.

Step 3. Perform Step 3 of Algorithm 3.3 to obtain the *h-sorted label sequences* L_1 and L_2 of S' and T, respectively.

Step 4. Group the labels of L_1 and L_2 by the following steps.

4.1 Group the labels of L_1 based on the h-feature values of the tile images in S', with each resulting group including the labels of a set of tile images having the same h-feature values.

4.2 Group the labels of L_2 based on the grouping of L_1 just obtained, resulting

in, say m, groups of labels, G_1, G_2, …, G_m, with each group G_i including the labels of a set of target blocks whose corresponding tile images have the same h-feature values.

Stage 2 --- embedding the secret message M.

Step 5. Generate the histogram H of the h-feature values of all the tile images in the resized secret image S'.

Step 6. Transform the message M to be embedded into a bit string M'.

Step 7. Perform the following steps to embed M' until all bits of M' are exhausted.

7.1 Select the smallest *unprocessed* h-feature value h_i in the range of 0 through 584 whose histogram value $H(h_i)$ is larger than or equal to *two* (with all h_i's regarded as *unprocessed* initially).

7.2 Take out the group G_i of labels in L_2 corresponding to the h-feature value h_i.

7.3 Take out the first two *unprocessed* labels l_1 and l_2 in G_i (with all labels in G_i regarded as *unprocessed* initially).

7.4 Switch the order of l_1 and l_2 in L_2 if the following two conditions are satisfied, assuming that the first *unembedded* bit in M is denoted as b:

A. $b = 0$ and $l_1 > l_2$;

B. b is 1 and $l_1 < l_2$

(i.e., the after-switching labels l_1' and l_2' are such that $l_1' < l_2'$ when $b = 0$; and $l_1' > l_2'$ when $b = 1$).

7.5 Repeat Steps 7.3 and 7.4 until G_i includes at most one label, which is left untouched.

7.6 Repeat Steps 7.1 through 7.5 if the bits of M' are not exhausted.

Step 8. Embed a string M' of 8 extra bits of zero as the *ending signal* of the input message M by Step 7 above.

Step 9. Fit the tile images of S' into the target blocks of T based on the one-to-one mappings from the labels of L_1 to those of the *re-ordered* L_2 with switched labels obtained in Steps 7 and 8 above (denoted as L_2' subsequently), and let the resulting image be denoted as T'.

Stage 3 --- dealing with unfilled target blocks and generating the desired mosaic image.

Step 10. Perform Step 5 of Algorithm 3.3 to fill each of the remaining unfilled target blocks if there is any.

Step 11. Based on two label sequences L_1 and L_2', generate a recovery sequence L_R according to the following steps.

11.1 Generate initially all the one-to-one mappings from the labels in L_1 to those of L_2'.

11.2 Sort all the labels in L_1 by their magnitudes with the smallest label being taken as the first, and do the same to the one-to-one mappings to re-order the labels in L_2' accordingly.

11.3 Take the re-order L_2' as L_R.

11.4 Transform L_R into a binary string.

Step 12. Perform the following steps to embed some extra data into image T'.

12.1 Embed the information of S', including its width $W_{S'}$ and height $H_{S'}$ as well as the size Z_t, into the first ten pixels of T' in a raster-scan order by the scheme of lossless LSB modification.

12.2 Compute *the number* N_P *of bits to be embedded for embedding* L_R by Equations (4.2) through (4.4).

12.3 Select an unprocessed tile image T_i (now fitted in the target image T) randomly by the secret key K (with all tile images in T regarded unprocessed initially).

12.4 Embed the first unembedded bit b_m of L_R into a pixel in T_i (with all bits in L_R regarded as unembedded initially), and in the meantime, decrease N_P by 1.

12.5 Repeat Step 12.4 if the pixels in T_i are not exhausted.

12.6 Examine N_P, and if it is nonzero, then go to Step 12.3; otherwise, continue.

Step 13. Generate the desired output image R by composing all the tile images fitted at their respective positions in T as an image.

4.2.2 Secret message extraction process

In the proposed secret message extraction process, first we have to recover the secret image. For this, with an inverse scheme of lossless LSB modification (i.e., by extracting LSB's from relevant pixels), we can get the recovery sequence L_R; and then retrieve accordingly the original secret image S. Also, by calculating the h-feature values of the original secret image, we can regain the h-feature values of the tile images, and sort them based on the values in order to get the h-sorted label sequence L_1.

As shown by Figure 4.3, because the recorded sequence L_R, though including only the labels of L_2 (see Step 11 of Algorithm 4.1), essentially specifies one-to-one mappings between the tile images and the target blocks, the h-sorted label sequence L_2 of the target blocks can be regained from corresponding mappings from L_1 to L_R. An example of retrieval of the sequence L_2 is given in Figure 4.4. With the histogram H of the h-feature values of all the tile images, we can group the labels of sequences L_1 and L_2, and then examine the orders of the labels of L_2 to extract the embedded secret message, in a way reverse to the message embedding process described in Algorithm

4.1.

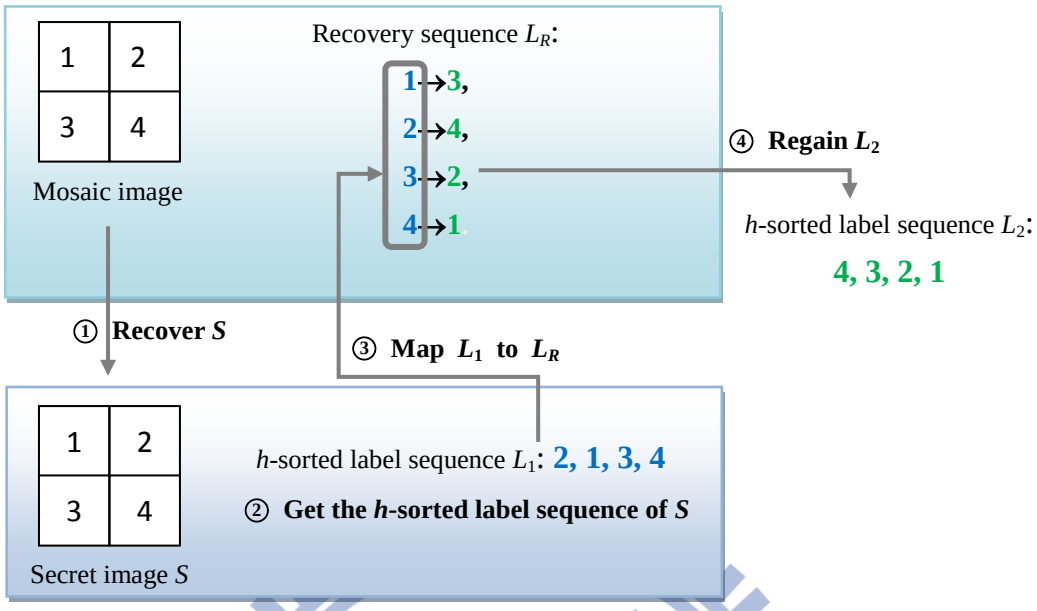


Figure 4.4 An illustration of the regaining of the h-sorted label sequence L_2.

The algorithms of secret image recovery and secret data extraction are described in detail as follows.

Algorithm 4.2: ***secret image recovery and secret message extraction.***

Input: a secret-fragment-visible mosaic image R, and a secret key K identical to that used in Algorithm 4.1.

Output: a recovered secret image S, and the secret message M supposedly embedded in R.

Steps:

Stage 1 --- retrieving the secret image S.

Step 1. Retrieve the parameter information of S, including W_S, H_S, and Z_t as mentioned in Algorithm 4.1, from the first ten pixels of R by an inverse scheme of lossless LSB modification.

Step 2. Perform the following steps to extract the recovery sequence L_R from R.

2.1 Get the number N_p of bits to be extracted for L_R by Equations (4.2) through

(4.4).

2.2 Select an *unprocessed* tile image T_i from R randomly by the key K (with all tile images in R regarded as unprocessed initially).

2.3 Extract an LSB from T_i in a raster-scan order by an inverse scheme of lossless LSB modifications, append it to a string B_i (initially empty), and in the meantime, decrease N_P by 1.

2.4 Repeat Step 2.3 if the pixels in T_i are not exhausted.

2.5 If $N_P \neq 0$, then go to Step 2.2; otherwise, continue.

2.6 Transform every L_B bits of B_i obtained in Steps 2.3 and 2.4 into a decimal value B_i', and add B_i' to a recovery sequence L_R as a label of it (L_R set empty initially).

Step 3. Compose the desired secret image S based on the sequence L_R by extracting the tile images fitted in R in order and placing them at correct relative positions.

Stage 2 --- regaining the h-sorted label sequences.

Step 4. Perform the following steps to get the h-sorted label sequence L_1 of the tile images (like that done in Algorithm 4.1).

4.1 Divide S into a sequence Q_S of blocks $S_1, S_2, \ldots, S_n$ as tile images based on size Z_t with i regarded as the label of S_i.

4.2 For each tile image, calculate its h-feature value based on Eq. (4.1).

4.3 Sort the h-feature values of all S_i, and re-order accordingly the blocks in Q_S to get a *re-ordered block sequences* Q_S'.

4.4 Get a new h-sorted label sequence L_1 from the labels of the re-ordered block sequences Q_S'.

4.5 Group the labels of L_1 based on the h-feature values of the tile images, with each resulting group including the labels of the tile images having the

same *h*-feature values.

Step 5. Perform the following steps to get the *h*-sorted label sequence L_2.

5.1 Get the re-ordered block sequence Q_T' of the target blocks by one-to-one-mapping the labels in sequence L_1 to those of L_R.

5.2 Get a new *h*-sorted label sequence L_2 from the labels of the re-ordered block sequence Q_T'.

5.3 Group the labels of L_2 based on the above grouping of L_1, resulting in m groups $G_1, G_2, \ldots, G_m$ with each group G_i including the labels of the target blocks whose corresponding tile images have the same *h*-feature values, h_i.

Stage 3 --- extracting the embedded secret message M.

Step 6. Generate the histogram H of the *h*-feature values of all the tile images in the recovered secret image S.

Step 7. Perform the following steps to extract the bits of secret message M until a presumably existing 8-bit ending signal is extracted.

7.1 Select the smallest *h*-feature value h_i in the range of 0 through 584 whose histogram value $H(h_i)$ is larger than or equal to two.

7.2 Take out the group G_i of labels in L_2 corresponding to h_i.

7.3 Take out the first two *unprocessed* labels l_1 and l_2 in G_i (with all labels in G_i regarded as unprocessed initially).

7.4 Examine the order of l_1 and l_2 in L_2 by the following rules to extract a hidden message bit b and append it to the end of a bit version of the message, denoted as D:

C. if $l_1 <= l_2$, then set $b = 0$;

D. if $l_1 > l_2$, then set $b = 1$.

7.5 Repeat Steps 7.3 and 7.4 until G_i includes at most one label (which is left

untouched).

7.6 Repeat Steps 7.1 through 7.5 if the 8-bit end signal is not extracted (i.e., if the last extracted 8 bits are not a sequence of 8 zero's).

Step 8. Transform every 8 bits of D into a hexadecimal number D' expressed as a UTF-8 code and then transform D' into characters as the desired secret message M.

4.3 Security Consideration

4.3.1 Issues of security of proposed method

As mentioned previously, users can recover a secret image and extract embedded messages from a secret-fragment-visible mosaic image by the proposed secret extraction process described by Algorithm 4.2 above. However, we do not want hackers to be able to extract these, too.

As we mentioned in Equations (4.2) through (4.4), N_T is the number of tile images, N_P stands for the number of bits to be embedded for embedding L_R, and Z_t stands for the size of a tile image. Each pixel in an image has three channels for embedding bits, and the lossless LSB modification needs two pixels to embed a bit [14], so the number N_Q of pixels required for embedding L_R is equal to $\left\lceil \frac{N_P \times 2}{3} \right\rceil$. Therefore, the number N_E of tile images required for embedding L_R is $\left\lceil \frac{N_Q}{Z_t} \right\rceil$. In the proposed creation process of secret-fragment-visible mosaic images, we use a secret key to select N_E tile images *randomly* for hiding labels of the recovery sequence L_R. Accordingly, the permutation of the selected tile images is $P_{N_E}^{N_T}$, leading to a probability of successful penetrating into the attacked image to extract relevant

information equal to $1/P_{N_E}^{N_T}$. In this study, we divide a secret image into numerous pieces to compose a secret-fragment-visible mosaic image. The value of N_T is larger than 40,000, and the value of N_E is around 32,000, so the probability is close to zero.

However, a wrong key may still have the chance to select some correct tile images in the steps of the recovery sequence extraction process (Step 2 of Algorithm 4.2). Therefore, an attacker may get parts of a secret image by using a wrong key. In this case, we must enhance the security measure for proposed method from the viewpoint of image steganography. A possible way for this goal proposed in this study is discussed next.

4.3.2 Proposed security enhancement measures

As shown by Figure 4.5, the proposed measure for security enhancement is to apply an exclusive-OR operation on a secret key and an embedded label of a recovery sequence. In the process of embedding the recovery sequence L_R (in Step 12 of Algorithm 4.1), we take out the first unembedded label l_i from L_R for hiding l_i into a tile image until all the labels of L_R are embedded. In more detail about this, first we transform the label l_i into a bit string l_i'. Next, we apply an exclusive-OR function on l_i' and a segment k_i of the secret key with a length equal to that of l_i. In this way, we can let the embedded value for l_i be more random. Even if the use of a wrong key leads to the selection of some correct tile images in the recovery sequence extraction process (Step 2 of Algorithm 4.2), the extracted label will be incorrect without the help of the right key for use in a reverse operation of the exclusive-OR function.

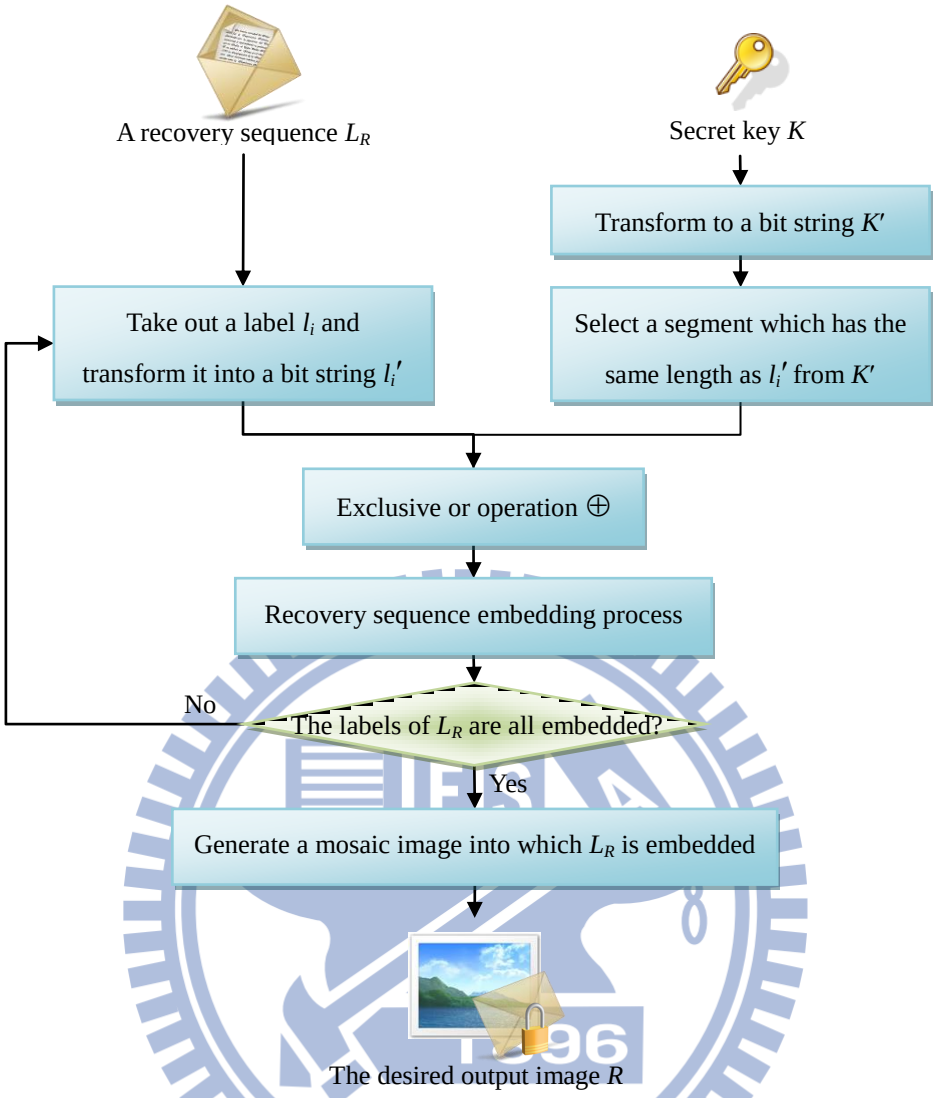


Figure 4.5 A flowchart of applying exclusive or operation on a secret message segment by a secret key.

4.4 Experimental Results

Some experimental images of proposed method are given in Figures 4.6 through 4.12. Figure 4.6(c) is the secret-fragment-visible mosaic image without secret message embedding. The average error at the block level in Figure 4.6(c) and Figure 4.8 (as the sum of all the Euclidean distances divided by the number of blocks) is 0.05, and the PSNR between them is 66.6. Obviously, the proposed information hiding method provides a good effect on covert communication, and the secret will be

transmitted imperceptibility.

If a user gives a wrong secret key in a secret extraction process, the extraction will be failed as shown by Figures 4.11 and 4.12. Because the secret extraction process needs the original secret image to calculate *h*-feature values, and use them to generate the 1-D *h*-colorscale histogram. With this information, the *h*-sorted label sequence L_2 of target blocks is ability to regain. The secret image will be retrieval only if the right key is used in the secret image recovery and secret message extraction process, like Figures 4.9 and 4.10.

4.5 Summary

While we calculating the *h*-feature values, the tile images which are classified to the same histogram have similar colors. By utilizing this characteristic of the secret-fragment-visible mosaic image creation process, we have proposed a novel information hiding method for covert communication techniques in this chapter. First, we transform secret data into binary code. Considering the security of embedded messages, we encode them with a secret key, and then switch the order of the labels of the *h*-sorted label sequence of target blocks by pairs based on the embedded bit. The difference between the mosaic image with secret embedding and without embedding is too small to observe by human beings. As a result, the proposed information hiding method is available for covert communication, and has good effects on it.

(a)

(b)

(c)

Figure 4.6 An example of secret-fragment-visible mosaic images. (a) A secret image. (b) A target image. (c) A secret-fragment-visible mosaic image without secret message embedding

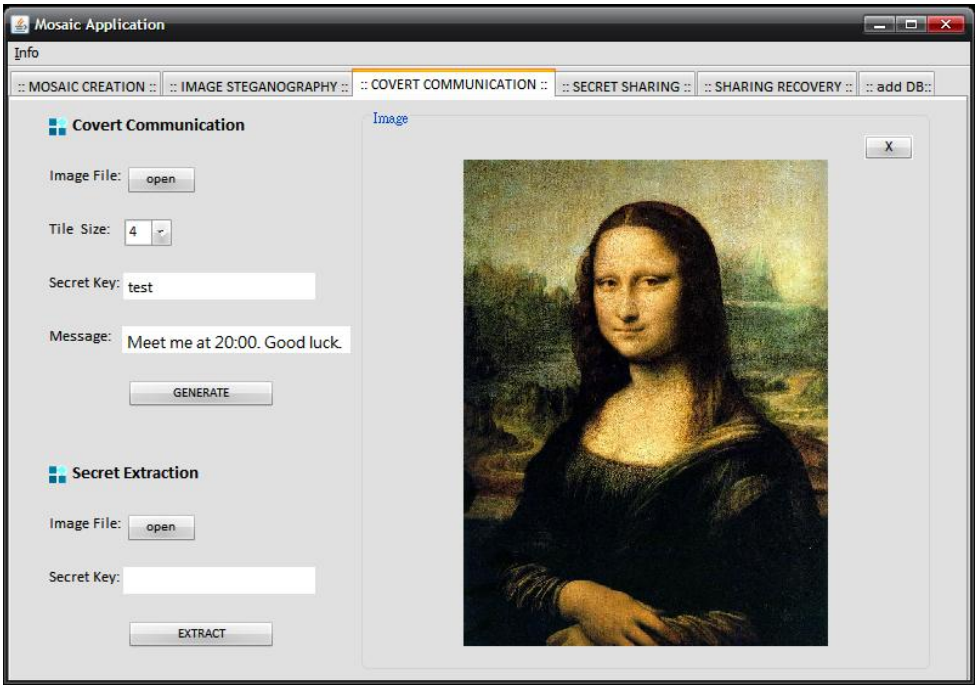


Figure 4.7 Embedding the secret message “Meet me at 20:00. Good luck.” with the secret key “test”.

Figure 4.8 A secret-fragment-visible mosaic image into which a secret message is hidden.

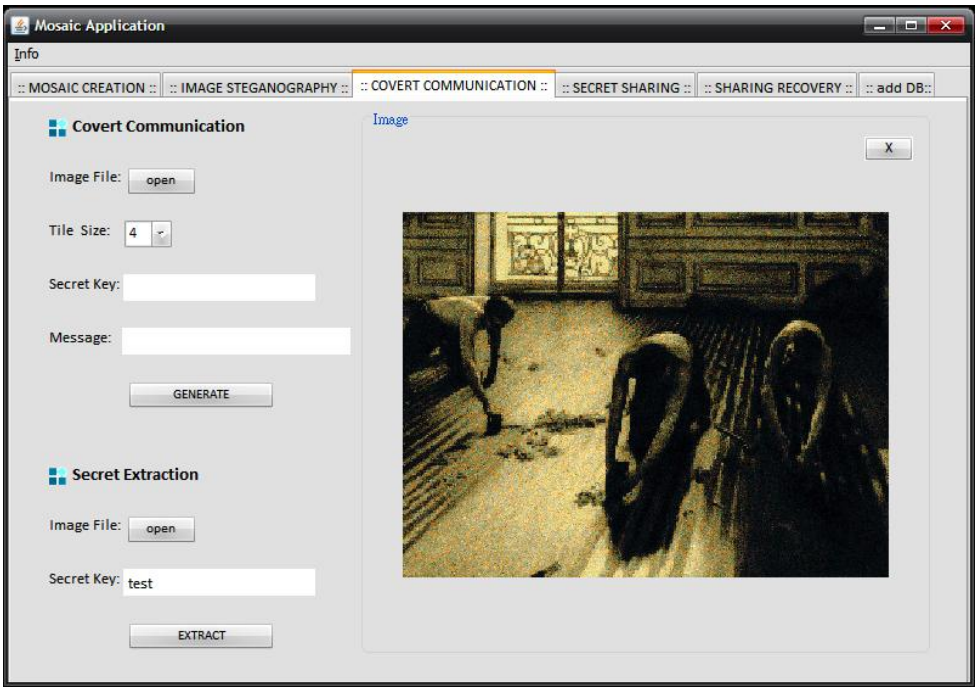


Figure 4.9 Extracting the secret message with the right secret key “test”.

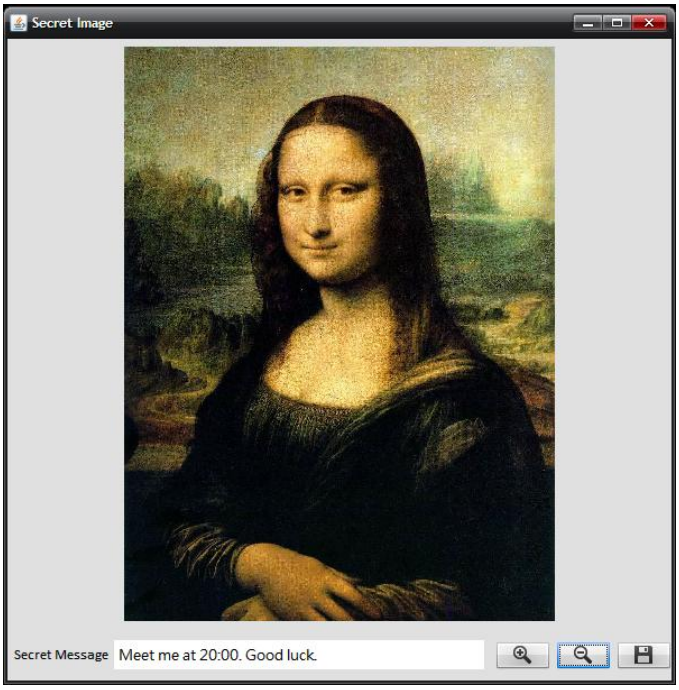


Figure 4.10 The resulting image and the extracted secret messages of Figure 4.8, which is recovered by the right key.

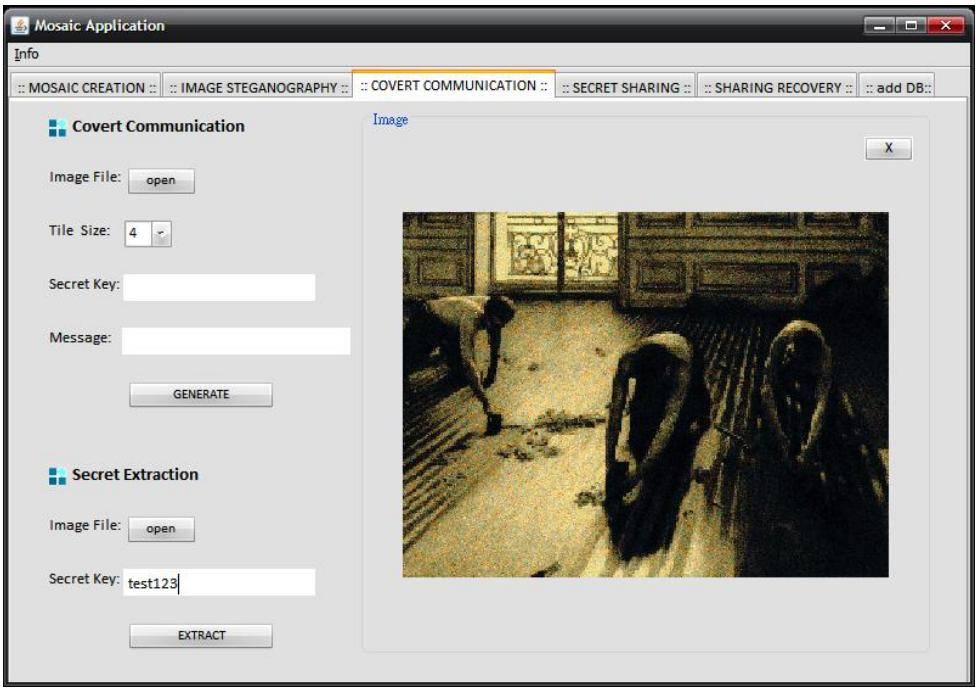


Figure 4.11 Extracting the secret message with the wrong secret key “test123”.

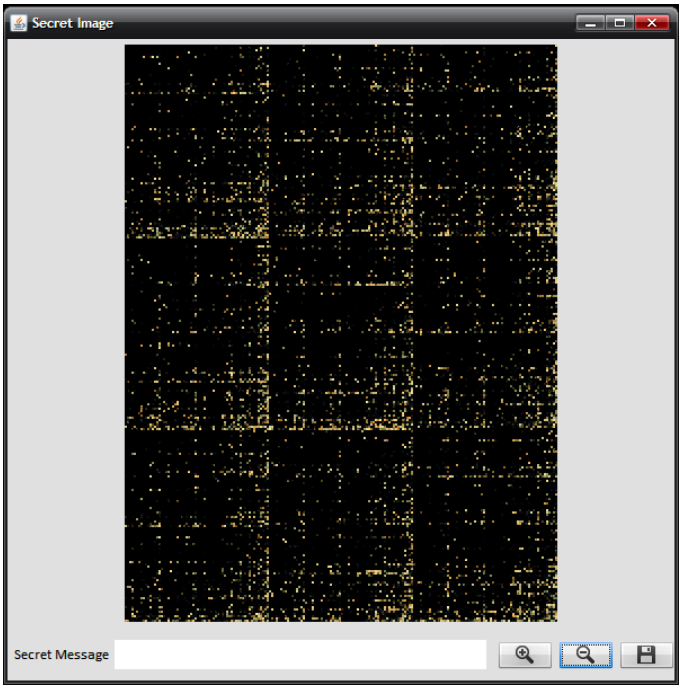


Figure 4.12 The resulting image and the extracted secret messages of Figure 4.10, which is recovered by a wrong key.

Chapter 5
Image Steganography via Secret-fragment-visible Mosaic Images

5.1 Idea of Proposed Method

In the former chapters, we have proposed a mosaic image creation process in order to generate a full-color secret-fragment-visible mosaic image. This scheme inspires us to apply the *grayscale* secret-fragment-visible mosaic image creation process for another kind of application — image steganography.

As we mentioned before, we can convert documents, such as e-mails, Microsoft WORD files, or PDF documents, into *grayscale* images through some commercially-available software packages (many downloadable on the Internet). By using this type of image and a secret key as the input to a corresponding secret-fragment-visible mosaic image creation process, we can generate a grayscale mosaic image in disguise. The transformed images keep the readability of the original documents by image fragments, and so a receiver can understand what the document content is after he/she recovers the secret image with the proposed method and the right secret key. But an outsider cannot.

For this, we extend the idea of the former chapter and modify the secret-fragment-visible mosaic image creation process proposed there in order to generate a grayscale secret-fragment-visible mosaic image. The related processes and algorithms will be described in the following sections.

5.2 Proposed Method for Image Steganography via Secret-fragment-visible Mosaic Images

5.2.1 Grayscale Secret-fragment-visible Mosaic Image Creation Process

As shown by Figure 5.1, the proposed grayscale secret-fragment-visible mosaic image creation process, not all the same as that proposed in the last chapter for generating full-color mosaic images, is composed of the major steps of database construction, mosaic image creation, and secret hiding. Furthermore, in order to select a sufficiently similar target image from the database, similarity measure computation again is required in the proposed mosaic image creation process. We will introduce them individually as follows.

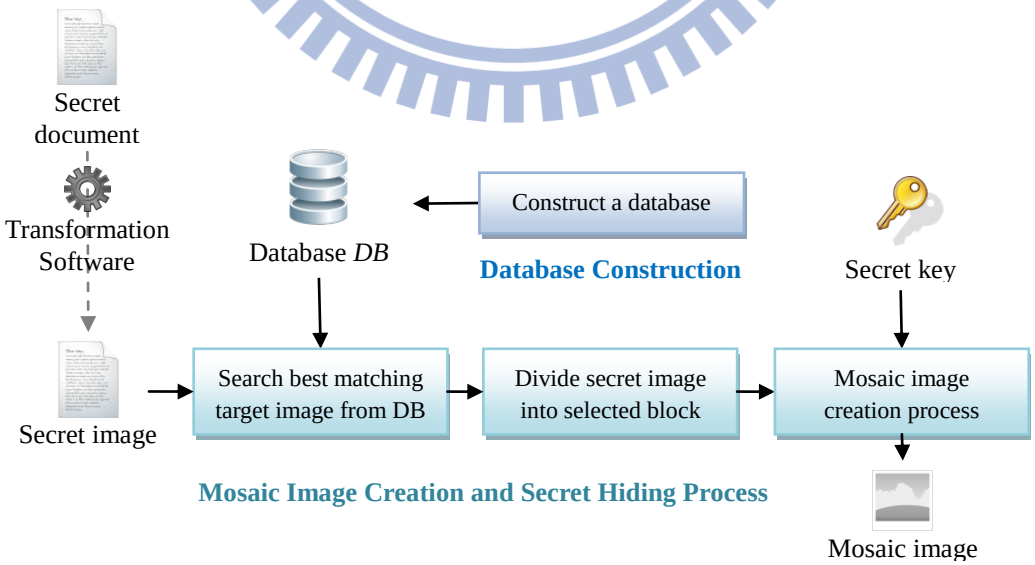


Figure 5.1 A flowchart of the proposed image steganography method through the use of grayscale secret-fragment-visible mosaic images.

A. Database construction

Images which are transformed from documents are colorless in this case now, so the values of the color channels of *R*, *G*, and *B* in such images are all the same. If we use a grayscale image as an input to the secret-fragment-visible mosaic image creation process, the results must be colorless, too. For this reason, in the database construction process to be described soon, we transform an input full-color image into a grayscale image by the following steps:

1. compute the average of the color values (*R*, *G*, *B*) of all the pixels in *C* as (R_C, G_C, B_C);
2. calculate the grayscale value g_C for *C* by the following equation:

$$g_C(R_C, G_C, B_C) = 0.177 \times R_C + 0.813 \times G_C + 0.011 \times B_C, \quad (5.1)$$

where the weights of the R_C, G_C, and B_C are set according to the parameters of the luminance in the color transformation from the *RGB* model to the *Yuv* one.

In addition, instead of using the 1-D *h*-colorscale histogram proposed in the last chapter, we use the distribution of the grayscale histogram to represent the content of a grayscale image here. The algorithm of grayscale candidate image database construction is shown as follows.

Algorithm 5.1: ***candidate image database construction.***

Input: a set of full-color images.

Output: a database *DB* of candidate images with size Z_c and their grayscale histograms, and the size Z_t of tile images for the candidate images in *DB*.

Steps:

Step 1. For each input image *I*, perform the following steps.

1.1 Resize and crop *I* to yield an image *D* of the pre-selected size Z_c.

1.2 Divide *D* into blocks of size Z_t.

1.3 For each tile image *C* of *D*, calculate the grayscale value g_C by Eq. (5.1) above.

1.4 Generate a histogram *H* of the values of the *grayscale values* of all the blocks in *D*.

1.5 Save *H* together with *D* into the desired database *DB*.

Step 3. If the input images are not exhausted yet, then go to Step 1; otherwise, exit.

B. Similarity measure computation and target image selection

In the database construction process described above, we generate a histogram *H* of grayscale values of all the blocks in an input image, and store it with the resized image into a candidate image database. Because the similarity measure computation is based on the information which is stored in the database, in this section we try to calculate the difference between the histogram of a secret image and that of each of the candidate images which are stored in the database in order to select a sufficiently similar target image for the mosaic image creation process. The detailed algorithm is described as follows.

Algorithm 5.2: ***Similarity measure computation.***

Input: a secret image *S*, a database *DB* of candidate images, and the sizes Z_t and Z_c of tile images and candidate images, respectively, mentioned in Algorithm 5.1.

Output: a target image *T* selected from *DB* with the largest content similarity to *S*.

Steps:

Step 1. Resize S to yield an image S' of size Z_c to become of the same size as that of the candidate images in *DB*.

Step 2. Divide S' into blocks of size Z_t, and perform the following steps.

2.1 For each block C of S' generated in Step 1, calculate the *grayscale* value g_C described by Eq. (5.1) and round off the result.

2.2 Generate a grayscale histogram $H_{S'}$ for S' from the computed grayscale values of all the blocks in S'.

Step 3. For each candidate image D with grayscale histogram H_D stored in *DB*, perform the following steps.

3.3 Compute an error e as the similarity measure between $H_{S'}$ and H_D by the following equation:

$$e = \sum_{m=0}^{255} \left| H_{S'}(m) - H_D(m) \right|; \tag{5.2}$$

where m stands for the value of the grayscale mentioned above.

3.4 Record the error e.

Step 4. If the images in *DB* are not exhausted, then go to Step 3; otherwise, continue.

Step 5. Select the image in *DB* which has the minimum error value and take it as the desired target image T.

C. Mosaic image creation and secret hiding process

After calculating the average grayscale value of each tile image, we can take it as a feature of the image. In the proposed mosaic image creation process, we use the grayscale histogram as a select function of a greedy algorithm for selecting the most similar target image.

In the proposed creation method, we divide a secret image into numerous tile images. Because the number of tile images is very large, and the grayscale values of

an image which is converted from a document usually has a distribution over 0 through 80, if we use only 256 bins for generating a grayscale histogram. Therefore, the histogram bins which range between 0 and 80 will have too many tile images. In a step of the mosaic image creation process which we will describe soon, elements in each bin of the histogram will be sorted. The running time of a sorting process is $O(n\log n)$, where n stands for the number of sorted member. Note that n is the number of tile images which are classified to the same bin in the sorting process of the proposed mosaic image creation process. The largest number of the histogram bin which ranges between 0 and 80 is around 2,500. According to our experimental experience, the sorting process will spend too much time. Considering the running time of the mosaic image creation process, we propose a new feature, called *k*-feature, for each block C in an image, denoted as k_C, which is described as follows:

$$k_C(\bar{g}) = 10 \times \bar{g}, \tag{5.3}$$

where $\bar{g}$ stands for the average grayscale value of all the pixels in a block. With this adjustment of the grayscale value, the number of bins is increased for 10 times but the number of tile images in each bin on the average will be decreased to about 1/10. Our experimental results show that the running time for the sorting is reduced greatly because of the mentioned reduction of the number n in each histogram bin. The algorithm of proposed mosaic image creation is given in detail as follows.

Algorithm 5.3: ***grayscale secret-fragment-visible mosaic image creation.***

Input: a grayscale secret image *S*, a database *DB*, and a selected size Z_t of a tile image.

Output: A secret-fragment-visible mosaic image *R*.

Steps:

Stage 1 --- embedding secret image fragments into a selected target image.

Step 1. Crop S to yield an image S' which is divisible by size Z_t.

Step 2. Perform following steps to select a target image T.

2.5 Select a candidate image by Algorithm 5.2 and denote it as T.

2.6 If the error e computed in Step 3.1 of Algorithm 5.2 is larger than a pre-selected threshold T_h, then enlarge the size of T for $\lceil e/T_h \rceil$ times.

Step 3. Perform the following steps to obtain a *block-label sequence* of each of S' and T.

3.6 Divide S' into a sequence $Q_{S'}$ of blocks S_1', S_2', …, S_n' as tile images based on size Z_t, where each block S_i' is said to have the *label i*.

3.7 Divide T into a sequence Q_T of blocks T_1, T_2, …, T_n as target blocks based on size Z_t, where the label of T_i is similarly defined.

3.8 For each tile image of S' and each target block of T, calculate the k-feature values of them based on Eq. (5.2).

3.9 Sort the k-feature values of all S_i' and all T_i, and re-order accordingly the blocks in $Q_{S'}$ and Q_T, respectively, to get two *re-ordered block sequences* $Q_{S'}'$ and Q_T'.

3.10 Get the new label sequences L_1 and L_2 from the re-ordered block sequences $Q_{S'}'$ and Q_T', and call them *k-sorted label sequences* of the resized secret image S' and the selected target image T.

Step 4. Fit the tile images into the target blocks based on the one-to-one mappings from the ordered labels of L_1 to those of L_2, thus completing the embedding of all the tile images in S' into all the target blocks of T.

Stage 2 --- dealing with unfilled target blocks.

Step 5. Perform the following steps to fill each of the remaining unfilled target blocks,

B, if there is any.

5.1 Compute the difference e' between the k-feature k_B of B and the k-feature k_A of each of the tile images, A, by the following equation:

$$e' = |k_B - k_A|. \tag{5.4}$$

5.2 Pick out the tile image A_o with the smallest error e', and then fill the tile image A_o into the target block B.

Stage 3 --- generating the desired mosaic image.

Step 6. Based on the two label sequences L_1 and L_2', generate a recovery sequence L_R according to Step 11 in Algorithm 4.1.

Step 7. Perform Step 12 in Algorithm 4.1 to embed some extra data into image T'.

Step 8. Generate the desired output image R by composing all the tile images fitted at their respective positions in T as an image.

5.2.2 Secret image recovery process

In a secret image recovery process, we have two input sets of data for the image retrieval. First is a secret-fragment-visible mosaic image, and the second is a user key. If the given user key is different from the one which is used in a grayscale secret-fragment-visible mosaic image creation process, the extraction of secret images will be unsuccessful. By using reversible LSB modification and the right key, we can recover the original secret image from a secret-fragment-visible mosaic image. The detail steps of the proposed secret image recovery process are given as follows.

Algorithm 5.4: secret image recovery and secret message extraction.

Input: a secret-fragment-visible mosaic image R, and a secret key K identical to that used in Algorithm 5.3.

Output: a recovered secret image S.

Steps:

Step 1. Retrieve the parameter information of S, including W_S, H_S, and Z_t as mentioned in Algorithm 4.1, from the first ten pixels of R by an inverse scheme of lossless LSB modification.

Step 2. Perform Step 2 in Algorithm 4.2 to extract the recovery sequence L_R from R.

Step 3. Compose the desired secret image S based on the sequence L_R by extracting the tile images fitted in R in order and placing them at correct relative positions.

5.3 Security Consideration

The security of the proposed method is based on the random selection of tile images for embedding labels of the recovery sequence in the mosaic image creation process. This process and Step 12 in Algorithm 4.1 are the same, so the issues of the two proposed methods are also the same. Here we also apply an exclusive-OR function on a secret key and each label of the recovery sequence in order to let the embedded value of a label be more random. This scheme can make the proposed method be more secure.

5.4 Experimental Results

Some experimental results of applying the proposed method for image steganography are shown by Figures 5.2 through 5.5. Figures 5.2(a), 5.4(a), 5.6(a) are the secret images of Figures 5.3, 5.5, and 5.7. Furthermore, Figures 5.2(b), 5.4(b), and 5.6(b) are the selected target images for the secret images. By using Figures 5.2(a) and a secret key “test”, we can generate a secret-fragment-visible mosaic image, as shown by Figures 5.8 and 5.9. We use Figure 5.3 as an input image to the secret image

recovery process. As shown by Figures 5.10 and 5.11, by using the right key “test,” the secret image is retrieved successfully. If we give a wrong key in the process of secret image recovery, like Figure 5.12, the extracted secret image will be like an insignificant noise, as shown by Figure 5.13.

(a) (b)

Figure 5.2 Input images. (a) A secret image which is converted from a document. (b) A target image.

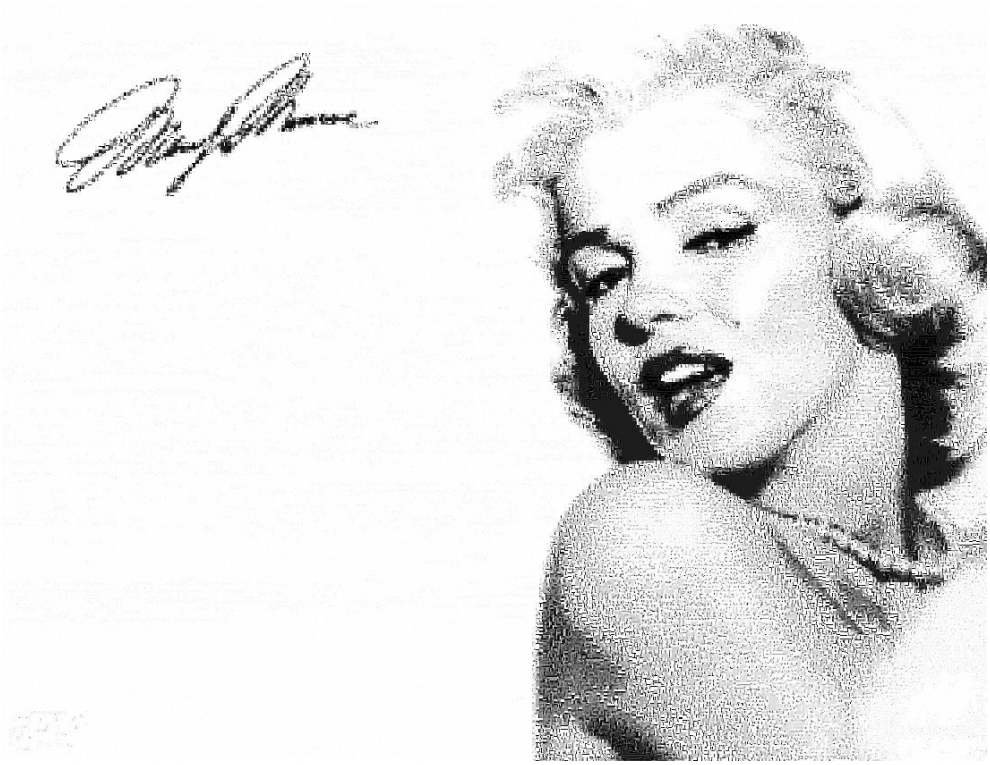


Figure 5.3 Resulting secret-fragment-visible mosaic image created with Figure 5.2(a).

Chapter 1
Introduction

1.1 Motivation and Background

1.1.1 Research Motivation

Over the last decade, the Internet has become more and more indispensable to people's daily life. Many social network services have been built on the Internet in recent years. Therefore, people can communicate with friends and share their information, such as interesting videos or images, through the services. Part of the information, such as intelligence data, web bank account passwords, and so on, is private and should be protected carefully. When the owners of such messages want to deliver them to other people, hackers may easily steal or tamper with the data contents. Accordingly, information hiding becomes an important issue nowadays.

On the other hand, artworks often arouse interests of people in daily life, especially artistic pictures which attract people's visual attention. Recently more and more visual arts are produced by computers, creating a new type of art, called *computer art*. An example is *mosaic image*, though it is a form of art existing for thousands of years, dating back to the days of Rome and Greece in the western world. Many information hiding techniques have been proposed in the past decade. Some of them were proposed via the use of images. The images, as carriers of information, were used just for the sole purpose of hiding information. If the images could be artworks like mosaic images, more attention of the people who rece who receive them

(a) (b)

Figure 5.4 Input images. (a) A secret image which is converted from a document. (b) A target image.

Figure 5.5 Resulting secret-fragment-visible mosaic image created with Figure 5.4(a).

Chapter 1
Introduction

1.1 Motivation and Background

1.1.1 Research Motivation

Over the last decade, the Internet has become more and more indispensible to people's daily life. Many social network services have been built on the Internet in recent years. Therefore, people can communicate with friends and share their information, such as interesting videos or images, through the services. Part of the information, such as intelligence data, web bank account passwords, and so on, is private and should be protected carefully. When the owners of such messages want to deliver them to other people, hackers may easily steal or tamper with the data contents. Accordingly, information hiding becomes an important issue nowadays.

On the other hand, artworks often arouse interests of people in daily life, especially artistic pictures which attract people's visual attention. Recently, more and more visual arts are produced by computers, creating a new type of art, called *computer art*. An example is *mosaic image*, though it is a form of art existing for thousands of years, dating back to the days of Rome and Greece in the western world. Many information hiding techniques have been proposed in the past decade. Some of them were proposed via the use of images. The images, as carriers of information, were used just for the sole purpose of hiding information. If the images could be artworks like mosaic images, more attention of the people who receive them

(a) (b)

Figure 5.6 Input images. (a) A secret image which is converted from a document. (b) A target image.

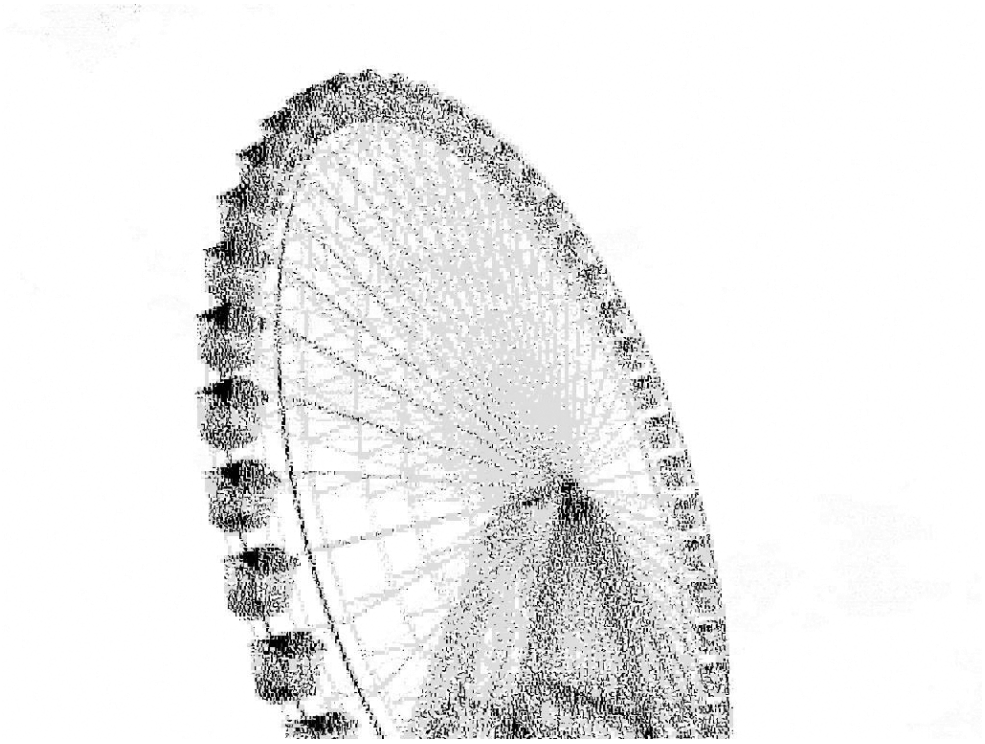


Figure 5.7 Resulting secret-fragment-visible mosaic image created with Figure 5.4(a).

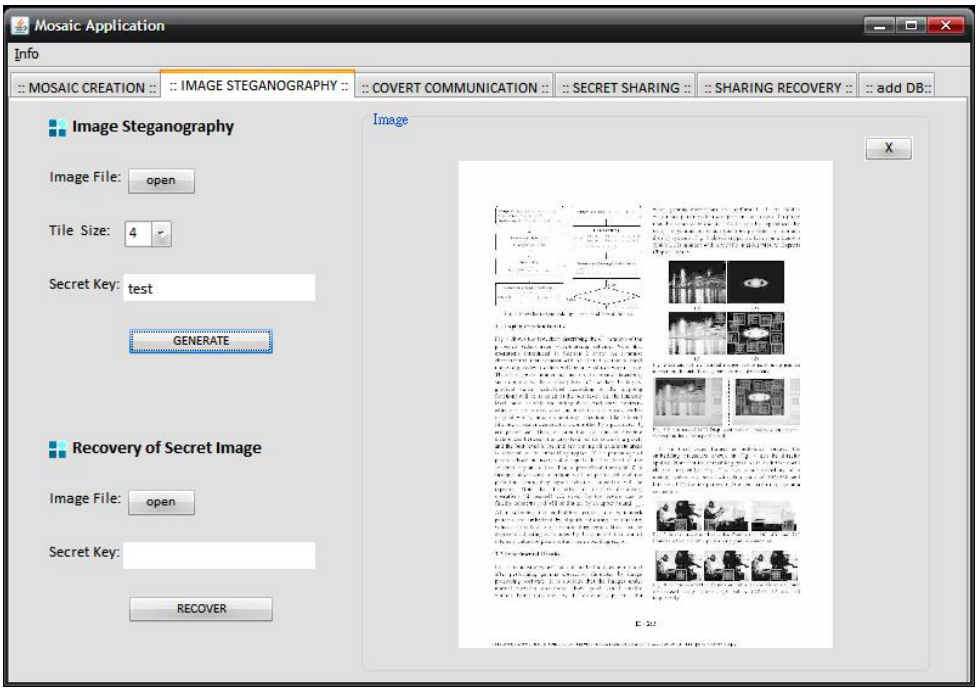


Figure 5.8 Embedding the secret image with the secret key “test”.

Figure 5.9 The resulting image which is created with a secret key “test”.

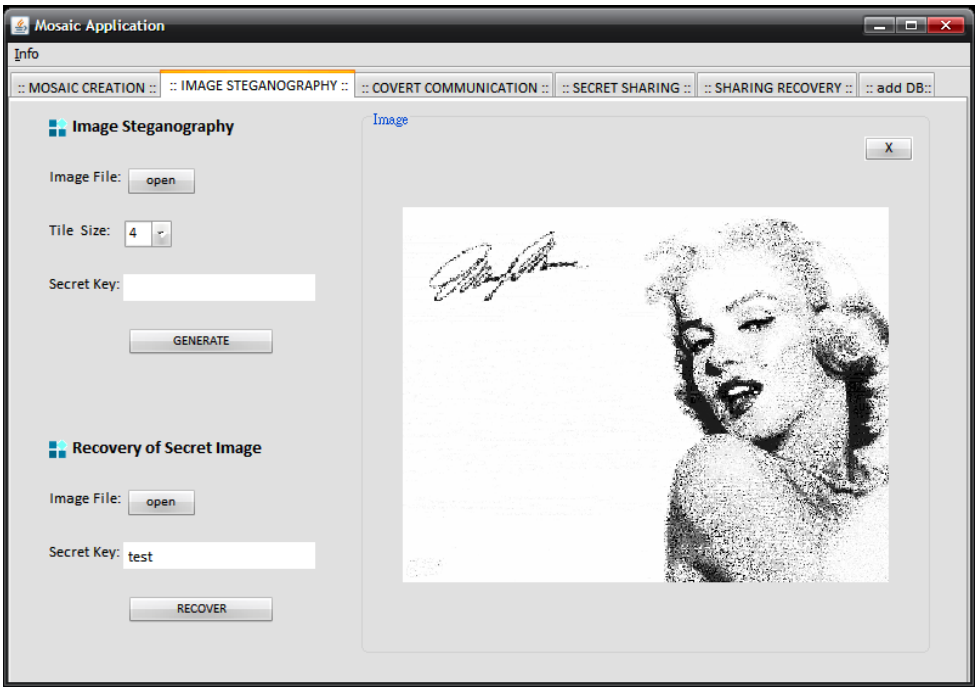


Figure 5.10 Extracting the secret image with the right key “test”.

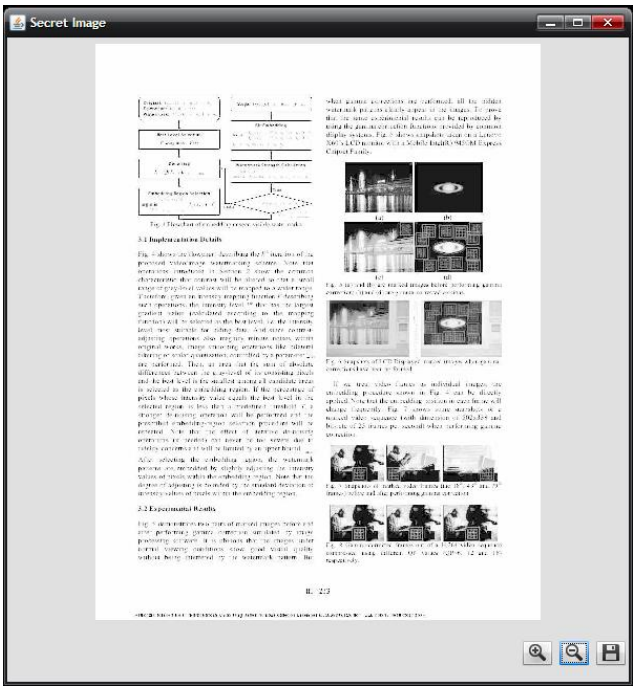


Figure 5.11 The resulting image and the extracted secret messages of Figure 5.4, which is recovered by the right key.

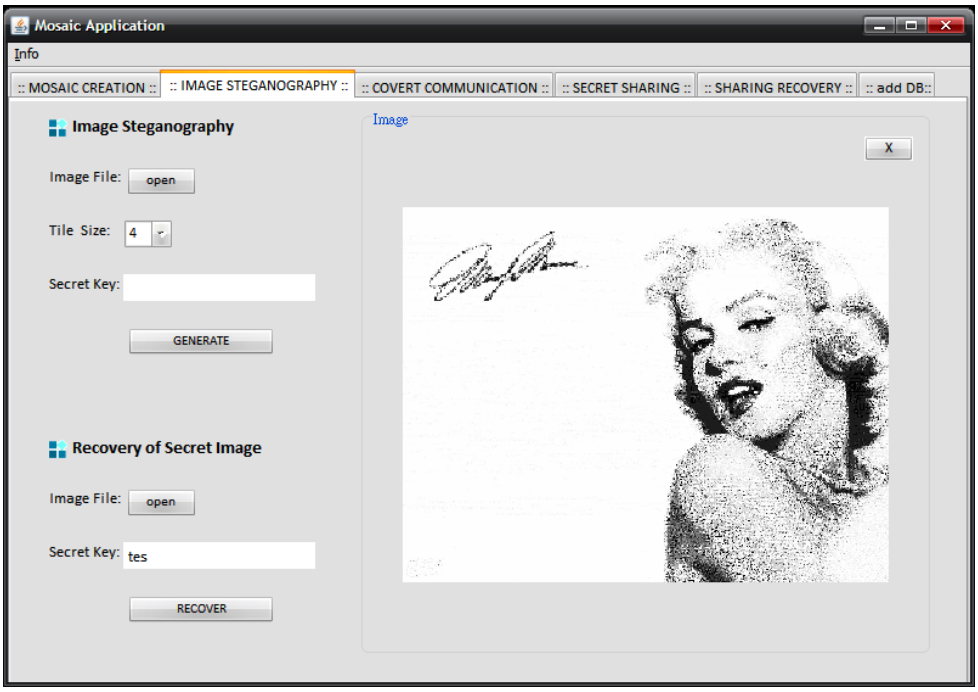


Figure 5.12 Extracting the secret image with the wrong key “tes”.

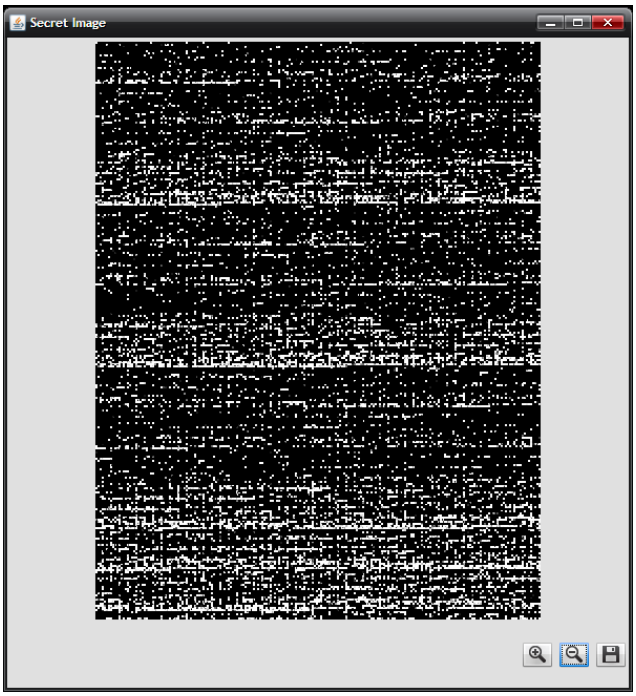


Figure 5.13 The resulting image and the extracted secret message of Figure 5.4, which is recovered with the wrong key.

5.5 Summary

In this chapter, we have proposed a novel method for image steganography. In this method, we presented a scheme of secret image hiding by using a secret image to create a secret-fragment-visible mosaic image. With a modified equation of the grayscale value, the *k*-feature of an image can be extracted and recorded into a candidate image database. Based on this feature, a new similarity measure was used to select the most similar target image for the secret-fragment-visible mosaic creation process. By recording the label of the target block which corresponds to every tile image, we can get a recovery sequence of labels. And by using the lossless LSB modification and a secret key to hide the sequence, a desired mosaic image is created. For security, a user key is used to randomize the data embedded in the target image. And only the use of the correct key can guarantee correct extraction of the secret image in the secret image recovery process.

In short, the proposed method indeed implements image steganography, and the experimental results have shown its feasibility for real applications.

Chapter 6
Secret Sharing via Secret-fragment-visible Mosaic Images

6.1 Idea of Proposed Method

The major idea of *secret sharing* is to distribute a secret message among a group of *sharers*. In the former chapters, we proposed methods for dividing a secret image into pieces, and then rearrange them to form a secret-fragment-visible mosaic image. If we use these pieces not just to compose only one mosaic image, but to distribute the pieces averagely among several target images to create several mosaic images, the scheme will be similar to secret sharing. Therefore, we propose in this study a method for sharing a secret image to form several secret-fragment-visible mosaic images, called *shares*, in order to implement *secret image sharing*. The method is described in this chapter.

In secret-fragment-visible mosaic image creation, the recovery sequence, which includes mappings between the labels of tile images and target blocks, is hidden in the resulting mosaic images by lossless LSB-modification. In the current investigation of secret image sharing, we cannot use this scheme anymore. The concept of recovering the shared secret adopted in this study is that the secret image is retrieved only if *all* the shares and the right key are collected together. Thus, if we use the same scheme of secret-fragment-visible mosaic image creation, even if the sharing participants do not gathered all together, the secret image will still be retrieved *partially*, contrary to the

just-mentioned concept. In order to avoid this problem, a solution we propose in this study is to divide the recovery sequence of each secret-fragment-visible mosaic image into several pieces and hide them in *other* images. The detail is described subsequently.

6.2 Proposed Secret Sharing Method

6.2.1 Algorithm for secret sharing

The procedure of the proposed secret sharing method is briefly illustrated in Figure 6.1. First, we select multiple candidate images as target images from the database by the criterion of selecting the most similar candidate images discussed in Chapter 3. Because we have to generate multiple mosaic images for the sharing participants, the selection of candidate images means to choose the top n most similar candidate images as the target images, where n stands for the number of sharing participants.

The algorithm proposed for similarity measure computation and target image selection is given as follows.

Algorithm 6.1: ***selection of n similar candidate images to be the target images.***

Input: a secret image S, a database DB of candidate images, a user-selected number n of sharing participants, and the sizes Z_t and Z_c of tile images and candidate images, respectively, mentioned in Algorithm 3.1.

Output: n target images, T_1 through T_n, which are selected from DB with the n largest similarities to S.

Steps:

Step 1. Resize S to yield an image S' of size Z_c which is the same as those of the

candidate images in *DB*.

Step 2. Divide S' into blocks of size Z_t, and perform the following steps.

2.1 Calculate and round off the *h*-feature value h_C for each block C of S' generated from Step 1 by Step 2.1 in Algorithm 3.2.

2.2 Generate a 1-D *h*-colorscale histogram $H_{S'}$ for S' from the values of the *h*-features of all the blocks in S'.

Step 3. For each candidate image D with 1-D *h*-colorscale histogram H_D stored in *DB*, perform the following steps.

3.5 Compute an error e as the similarity measure between $H_{S'}$ and H_D by Eq. (3.4).

3.6 Record the error e.

Step 4. If the images in *DB* are not exhausted, then go to Step 3; otherwise, continue.

Step 5. Select the first n candidate images in *DB* which have the n minimum error values, and take them as the desired target images T_1 through T_n.

After selecting the target images, we fit the tile images into the target blocks in each target image. In the fitting process, we let the target images to take turns *randomly* (with the order decided by a secret key), each time picking a tile image and fitting it into a randomly-selected target image. In the meantime, we record the *h*-feature value of each selected tile image for every target image.

Because we have to hide the recovery sequence of each target image T_i into other target images, we have to assign unique labels to them to generate a re-ordered label sequence of the target images based on their contents. This label sequence is also needed in the secret image recovery process later on. For create this sequence, according to the recorded *h*-feature values, we can generate the 1-D *h*-colorscale histogram for each T_i. By utilizing the largest value $H(h_i)$ of bins in each T_i, we can

re-order the images T_1 through T_n, and get the new labels T_1', T_2', …, T_n' of them based on the order.

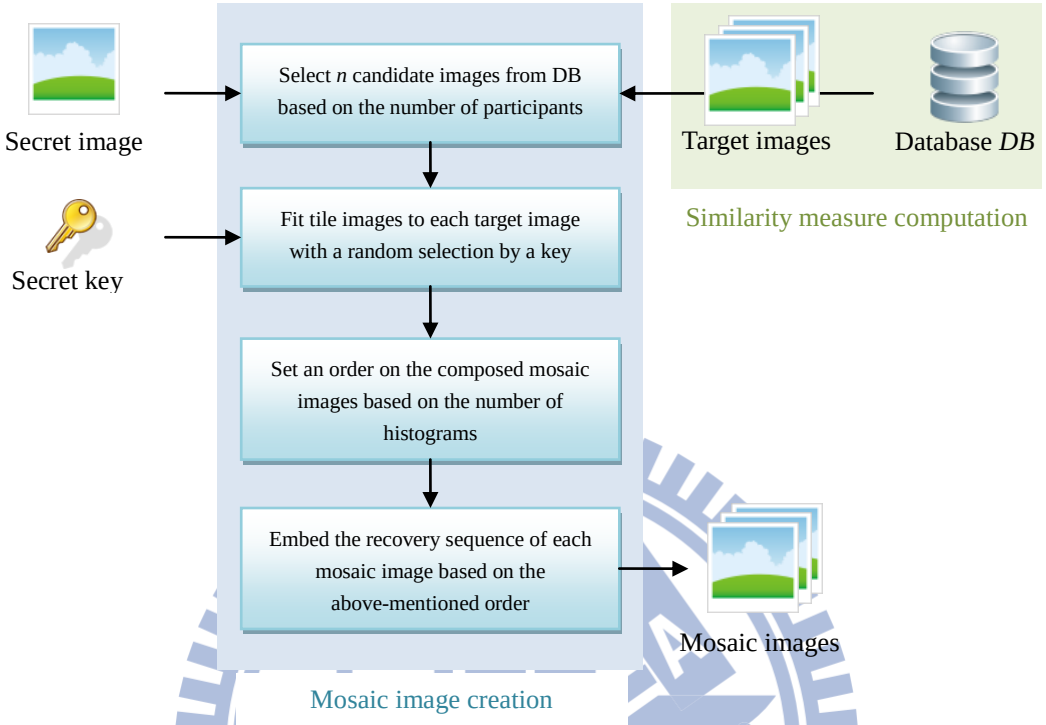


Figure 6.1 A flowchart of the secret sharing process via secret-fragment-visible mosaic images.

For example, suppose that a secret image is shared into two secret-fragment-visible mosaic images, T_A and T_B, and their largest values of bins are $H(h_A)$ and $H(h_B)$, respectively. We will assign new labels to T_A and T_B by the following rule:

1. If $H(h_A) > H(h_B)$, then we will assign a smaller label $T_1' = 1$ to T_A, and a larger one $T_2' = 2$ to T_B;
2. If $H(h_A) < H(h_B)$, then we will assign a smaller label $T_1' = 1$ to T_B, and a larger label $T_2' = 2$ to T_A.

However, the value of $H(h_i)$ in each mosaic image may be *repeated* so that the above re-ordering of some T_i's may not be unique. In this case, we will apply the following steps to break this tie, say for T_A and T_B:

1. if $h_A > h_B$ (not $H(h_A) > H(h_B)$ now), then we assign a smaller label T_1' to T_A, and a larger one T_2' to T_B;
2. if $h_A < h_B$, then we assign a smaller label T_1' to T_B, and a larger one T_2' to T_A;
3. if $h_A = h_B$, then we will select randomly a block which belongs to the nearest bin of h_A by a user key K, and change its h-feature value to be h_A in order to break the tie and assign a smaller label T_1' to T_A, and another larger one T_2' to T_B.

With these rules, we can get the new labels of target images, and use them in the embedding process of recovery sequences.

After finishing the fitting process, we can generate n images, each of which is assigned a label T_i' according the above-mentioned schemes. To generate a new recovery sequence L_{Ri}' of a certain T_i', first, we divide all the recovery sequences of the resulting mosaic images into $n - 1$ segments, and then take out one segment out of each of all the recovery sequences *except the sequence* L_{Ri} *itself*. Then, we combine the selected segments based on the labels of their mosaic images. For example, if there are three mosaic images T_1' through T_3' with the original recovery sequences L_{R1} through L_{R3}, and we divide each recovery sequence into 2 segments. To generate the new recovery sequence L_{R1}' of T_1', we take out the first uncombined segments of L_{R2} and L_{R3}, and combine these segments into a new recovery sequence L_{R1}', as shown by Figure 6.2.

With the basic concepts discussed as above, the detailed algorithm now can be given as follows.

Algorithm 6.2: ***Secret sharing process.***

Input: a secret image, a secret key K, a candidate image database DB, a selected size

Z_t of a tile image, and a user-selected number n of sharing participants.

Output: n share secret-fragment-visible mosaic images, R_1 through R_n.

Steps:

Stage 1 --- selecting multiple candidate images for a secret image from a database.

Step 1. Crop S to yield an image S' with a size *divisible* by Z_t.

Step 2. Select n candidate images by Algorithm 6.1 as the target images, T_1 through T_n.

Step 3. Perform the following steps to obtain a *block-label sequence* of S' and the h-feature value of each target block in T_i of T_1 through T_n.

3.11 Divide S' into a sequence $Q_{S'}$ of blocks S_1', S_2', …, S_m' as tile images based on size Z_t, where each block S_i' is said to have the *label* i.

3.12 Divide each image of T_1 through T_n, denoted as T_i, into target blocks based on size Z_t.

3.13 For each tile image of S' and all the target blocks of each T_i, calculate the h-feature values of them based on Eq. (3.3).

3.14 Sort the h-feature values of all S_i', and re-order accordingly the blocks in $Q_{S'}$ to get a *re-ordered block sequences* $Q_{S'}'$.

3.15 Get the new label sequences $L_{S'}$ from the re-ordered block sequences $Q_{S'}'$, and call it *h-sorted label sequences* of the resized secret image S'.

Stage 2 --- fitting all the tile images evenly into the selected target images.

Step 4. Group the labels of $L_{S'}$ to n groups based on the order of labels in $L_{S'}$, resulting in, say p, groups of labels, G_1, G_2, ..., G_p, with each group including the labels of a set of tile images of S'.

Step 5. Select an unprocessed group G_i of labels.

Step 6. Select an unprocessed target image T_i from T_1 through T_n by the key K.

Step 7. Perform the following steps to find the most similar tile image A_i for a target

block B_i of T_i.

7.1 Take out an unfitted tile image A_i whose label is in G_i.

7.2 Calculate the error e_i between A_i and a unfilled target block B_i of T_i by the following equation:

$$e_i = |h_A - h_B|, \tag{6.1}$$

where h_A stands for the h-feature value of A_i, and h_B stands for the h-feature value of B_i.

7.3 Record the error e_i.

7.4 Repeat Steps 7.1 through 7.3 until the unfilled target blocks are all processed.

Step 8. If the *unfitted* tile images whose labels are in G_i are all processed, continue; otherwise, go to Step 6.

Step 9. Fit the tile image A_i to the target block B_i whose error e_i with respect to A_i is the smallest, and record the labels of A_i and B_i into a recovery sequence L_{Ri}.

Step 10. If the tile images whose labels are in G_i are all fitted, continue; otherwise, go to Step 6.

Stage 3 --- dealing with unfilled target blocks.

Step 11. Generate the 1-D h-color histograms, H_1, H_2, …, H_n, from the h-feature values of tile images and unfilled target blocks of each *semi-finished* mosaic image.

Step 12. Get the largest value of $H(h_i)$ of each target image T_i, and generate a sequence Q_H of $H(h_i)$.

Step 13. Perform the following steps to get a re-ordered sequence Q_H' of $H'(h_i)$ which comes from reordering of the sequence Q_H of $H(h_i)$.

13.1 Sort the numbers, $H(h_1)$, $H(h_2)$, …, $H(h_n)$, of Q_H based on the above-mentioned rules.

13.2 Generate a new sequence Q_H' of $H'(h_i)$ from the re-ordered Q_H.

Step 14. Perform the following steps to fill each of the remaining unfilled target blocks, C, for each target image.

14.1 Compute the difference e' between the h-feature h_C of C and the h-feature h_D of each of the tile images, D, by the following equation:

$$e' = |h_C - h_D|. \tag{6.2}$$

14.2 Pick out the tile image D_o with the smallest error e' and compare e' with another pre-selected threshold T_h in the following way:

A. if $e' < T_h$, then fill the tile image D_o into the target block C;

B. if $e' \geq T_h$, fill the average R, G, and C values of all the pixels in C into C.

Stage 4 --- hiding the recovery sequence of each mosaic image.

Step 15. Get new labels for T_1 through T_n based on the order of the magnitudes of their values of $H(h_i)$ in Q_H', resulting in the new labels, T_1' through T_n'.

Step 16. Embed the information of S', including its width $W_{S'}$ and height $H_{S'}$ as well as the size Z_t, into the first ten pixels of T_1' through T_n' in a raster-scan order by the scheme of lossless LSB modification.

Step 17. Perform the following steps to generate a new recovery sequence L_{Ri}' from the recorded recovery sequence L_{Ri} of each mosaic image T_i'.

17.1 Divide the labels of every recovery sequence L_{Ri} into $n - 1$ segments, resulting in the segments L_{i1} through $L_{i(n-1)}$.

17.2 Take out all the target images except those of T_i', resulting in a group G_i of A_1 through A_{n-1}.

17.3 Pick out the first uncombined segment of L_{i1}, L_{i2}, …, $L_{i(n-1)}$ in each mosaic image in G_i based on the order of T_i', resulting in a sequence Q_L of L_{i1}', L_{i2}', …, $L_{i(n-1)}'$.

17.4 Combine the segments L_{i1}', L_{i2}', …, $L_{i(n-1)}'$ of the sequence Q_L into a new recovery sequence L_{Ri}' for T_i'.

17.5 If the new recovery sequences of all the target images are generated, continue; otherwise, go to Step 17.2.

Step 18. Embed the recovery sequence L_{Ri}' of each T_i' by Step 12 in Algorithm 4.1.

Step 19. Generate the desired output image R_1 through R_n by hiding the recovery sequences into T_1' through T_n'.

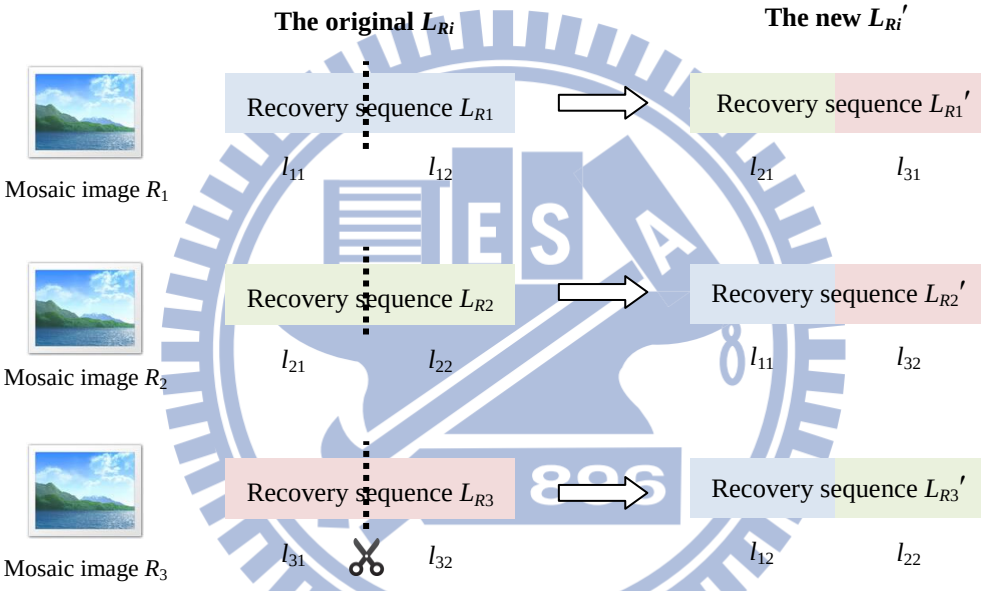


Figure 6.2 An illustration of the combination of the new recovery sequence of each sharing mosaic image.

6.2.2 Algorithm of proposed secret recovery process

Whether or not a secret image is recovered successfully is based on the completeness of sharing participants and the secret key. If the sharing participants are not the same as those participating the original secret sharing process, the secret image should not be retrieved. In the proposed secret sharing process, we assign new labels to the target images based on their largest values of $H(h_i)$, and generate new

recovery sequences which are then embedded into the resulting mosaic images according to the order of the new labels. Therefore, in the proposed secret recovery process, the secret image will be regained only if all the original shares are collected together. The detailed algorithm is described as follows.

Algorithm 6.3: ***secret image recovery.***

Input: n shared secret-fragment-visible mosaic images R_1 through R_n, and a secret key K identical to that used in Algorithm 6.2.

Output: a recovered secret image S.

Steps:

Stage 1 --- regaining the re-ordered label sequence of sharing participants.

Step 1. Retrieve the parameter information of S, including W_S, H_S, and Z_t as mentioned in Algorithm 6.2, from the first ten pixels of anyone of the shared mosaic images by an inverse scheme of lossless LSB modification.

Step 2. Perform the following steps to get the sequence Q_H of largest values $H(h_i)$ of R_1 through R_n.

2.1 Divide R_1 through R_n into blocks based on the size of Z_t.

2.2 Calculate the h-feature value of each block of R_1 through R_n, respectively.

2.3 Generate 1-D h-color histograms, H_1, H_2, …, H_n, from the h-feature values of blocks of R_1 through R_n, respectively.

2.4 Get the largest value $H(h_i)$ of each R_i, where $i = 1, 2, \ldots, n$.

2.5 Generate a sequence Q_H of the values, $H(h_1)$, $H(h_2)$, …, $H(h_n)$, in order.

Step 3. Perform the following steps to get the re-ordered sequence Q_H'.

3.1 Sort the values $H(h_i)$ of Q_H based on the rules which are mentioned in Section 6.2.1.

3.2 Generate a new sequence Q_H' of $H'(h_i)$ from the re-ordered Q_H.

Step 4. Assign new labels to R_1 through R_n based on the order of their $H'(h_i)$ in Q_H', resulting in the new labels R_1' through R_n'.

Stage 2 --- extracting segments of the recovery sequence of each mosaic image.

Step 5. Extract the recovery sequence $L_{Ri'}$ of R_i', where i = 1, 2, …, n, by performing Step 2 in Algorithm 4.2 on each R_i'.

Step 6. Perform the following steps to regain the original recovery sequence $L_{Ri'}$ of each R_i'.

6.1 Select the first unprocessed mosaic image R_i' in the sequence L_M.

6.2 Generate a group G_R which is composed of the original recovery sequences, L_1, L_2, …, L_{n-1}, of all the mosaic images except R_i'.

6.3 Divide $L_{Ri'}$ into n – 1 segments l_1, l_2, …, l_{n-1}.

6.4 Put each l_i of l_1 through l_{n-1} into the corresponding recovery sequence L_1 through L_{n-1}.

6.5 If all the mosaic images are processed, then continue; otherwise, go to Step 6.1.

Stage 3 --- retrieving the secret image S based on the recovery sequences.

Step 7. Compose the desired secret image S based on each recovery sequence $L_{Ri'}$ of a mosaic image R_i' by extracting the tile images fitted in R_i' in order and placing them at correct relative positions.

6.3 Security Consideration

The security of the proposed method is also based on the random selection of tile images for embedding the labels of the new recovery sequence which is generated from the combination of the segments of recovery sequences of other resulting images as shown by Figure 6.2 in the mosaic image creation process. This process is a repeat

of Step 12 in Algorithm 4.1 for all the mosaic images, so the issues of the two proposed methods are the same.

Therefore, here we also apply an exclusive-OR function on a secret key and each label of the recovery sequence in order to let the embedded value of a label be more random. This scheme can make the security of the proposed method be firmer.

6.4 Experimental Results

Some experimental images are given in Figures 6.3 and 6.4. Figures 6.3(a) is a secret image, and we applied the proposed method on it to share this secret to 2 sharers. Figure 6.4(a) is another secret image which is shared to create three mosaic images, Figures 6.4(e), 6.4(f), and 6.3(g). The number of tile images of a secret image is limited, and they are distributed among the shares. Therefore, the larger the number of shares is, the more similar the resulting images are. An example is given in Figures 6.4(g) and 6.6. They were both created from Figure 6.4(a). Because the share number of Figure 6.4(g) is 3, and the share number of Figure 6.6 is 5, Figure 6.6 is more similar to the target image, Figure 6.4(d).

As shown by Figures 6.5 and 6.6, we use a secret key “test” to create five secret-fragment-visible mosaic images. In the secret recovery process, with all the sharing participants’ shares and the right key, the correct secret image will be regained, as shown by Figures 6.7 and 6.8. The experimental results of applying the proposed secret recovery method with incorrect shares are shown in Figures 6.9 and 6.10, from which we can see the recovered image is erroneous.

6.5 Summary

In this chapter, we have proposed a novel secret sharing method though the use

of secret-fragment-visible mosaic images as shares. We use the tile images of a secret image to compose multiple secret-fragment-visible mosaic images in order to achieve the goal of secret sharing. In the process of selecting target images, multiple target images for a secret image based on the number of shares are selected. By calculating the errors between the secret image and candidate images in a database, we get the top *n* most similar candidate images as target images for the subsequent mosaic creation process.

In order to disperse tile images randomly and evenly, selected target images take turns randomly to pick appropriate tile images controlled by a secret key. While generating the results, we can get 1-D *h*-colorscale histograms of these images. With this information, the largest value of the 1-D *h*-colorscale histogram of each mosaic image can be obtained. By comparing these values and its *h*-feature value, we can assign new ordering labels to the target images, and use them to create new recovery sequences, resulting in a property of secret sharing that the secret can be extracted successfully only if all the participants gather together. Through these related procedures, we can implement a method for secret sharing. And the experimental images show the feasibility of it.

(a)

(b)

(c)

(d)

(e)

Figure 6.3 An example of the proposed secret sharing method. (a) A secret image. (b) A selected target image. (c) Another selected target image. (d) A shared secret-fragment-visible mosaic image. (e) Another shared secret-fragment-visible mosaic image.

(a)

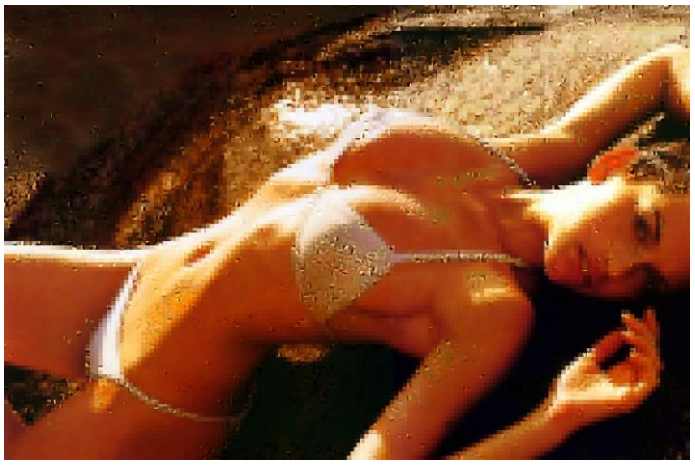
(b)

(e)

(c)

(f)

(d)



(g)

Figure 6.4 An example of the proposed secret sharing method. (a) A secret image. (b) A selected target image. (c) A second selected target image. (d) A third selected target image. (e) A shared secret-fragment-visible mosaic image. (f) A second shared secret-fragment-visible mosaic image. (g) A third shared secret-fragment-visible mosaic image.

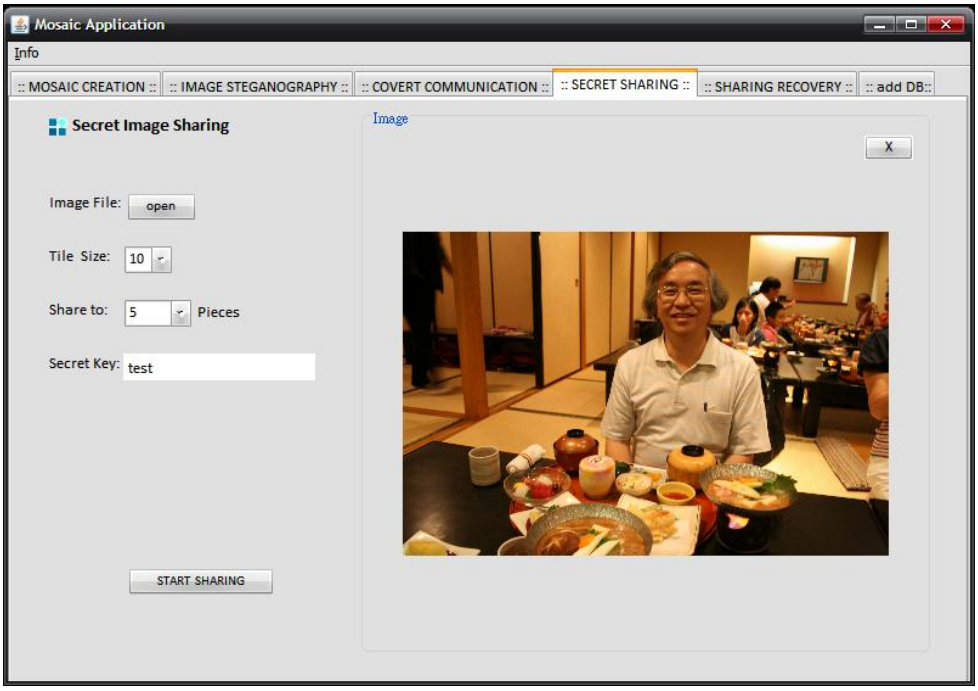


Figure 6.5 Sharing the secret image to 5 pieces, which is created with the secret key “test”.

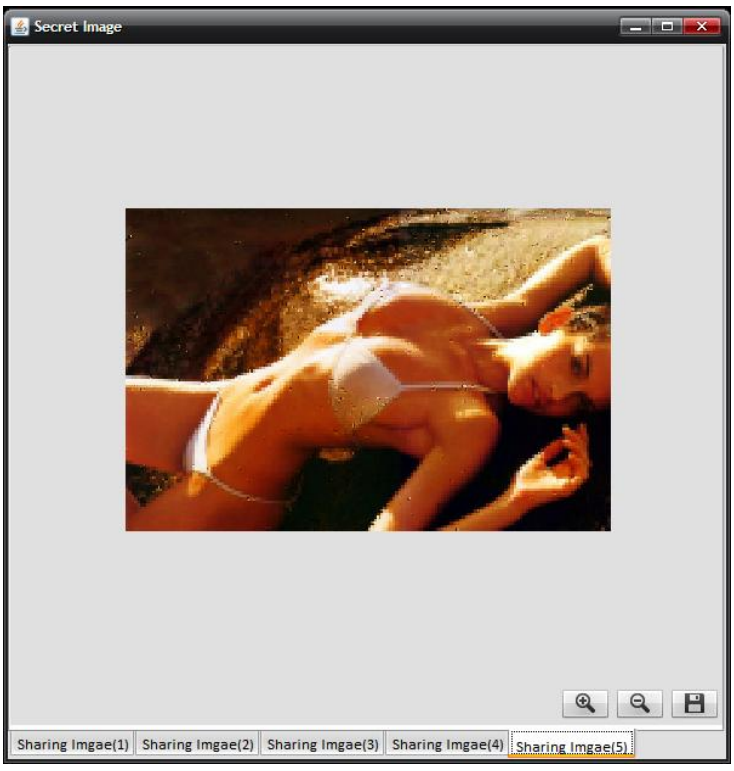


Figure 6.6 The resulting images of Figure 6.5.

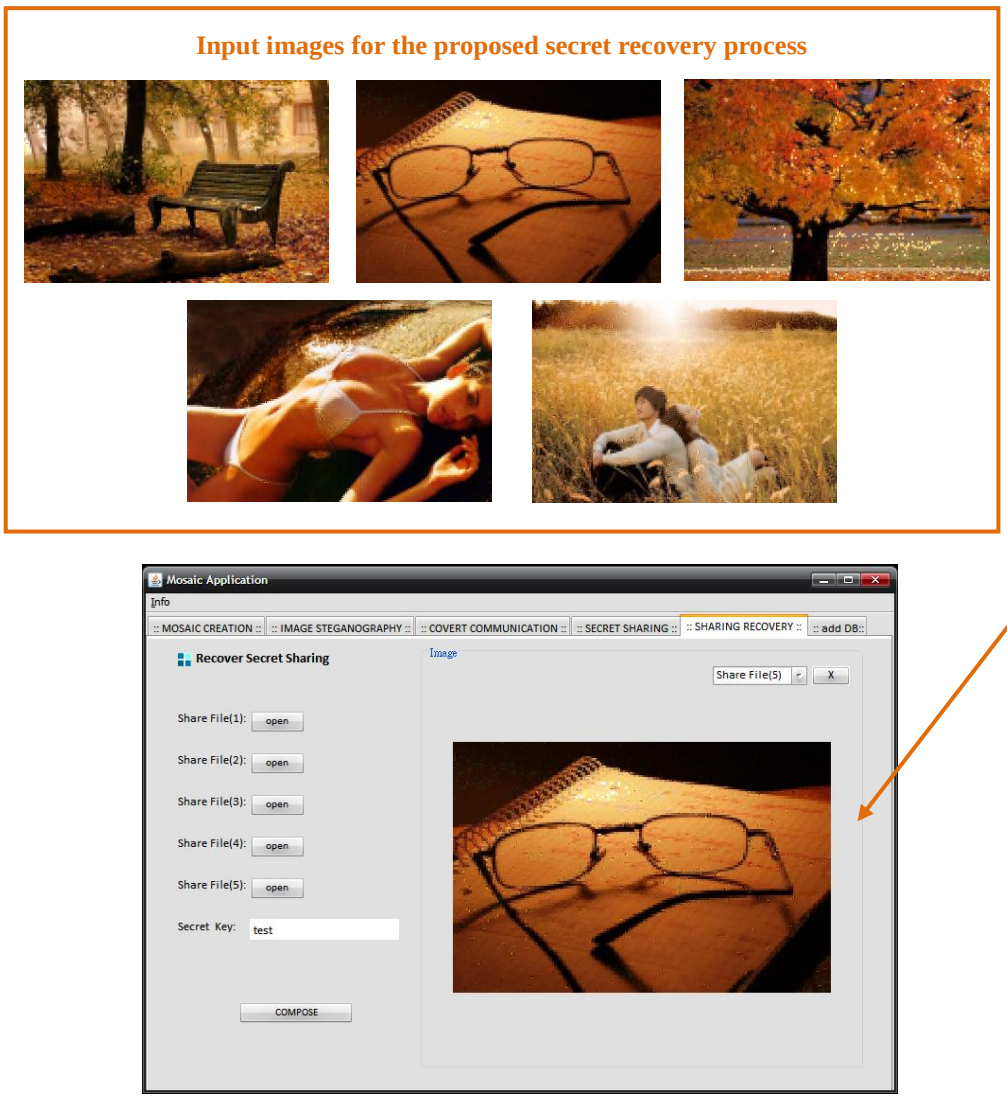


Figure 6.7 Recovery of the secret image with all the sharing participants and the right secret key "test".

Figure 6.8 The recovered images of Figure 6.7.

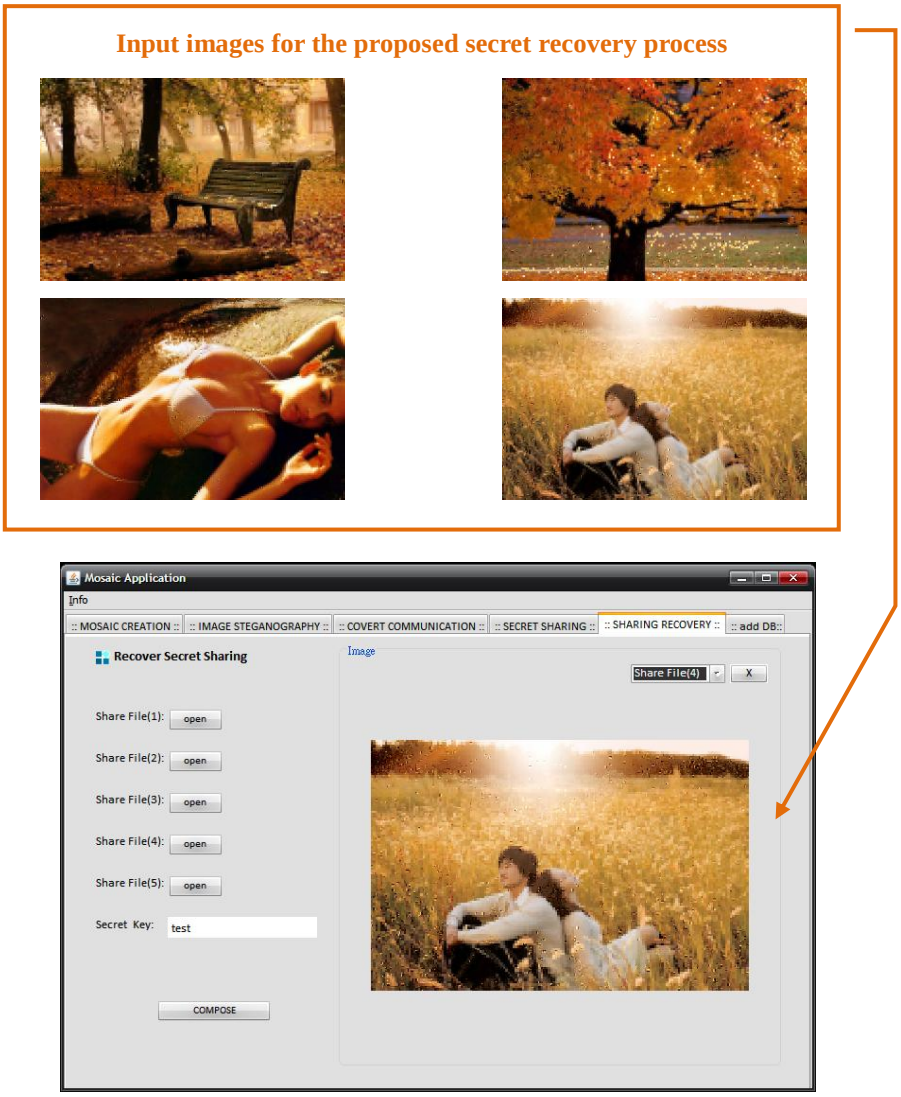


Figure 6.9 Using part of shares and the right key "test" for recovering the secret image.

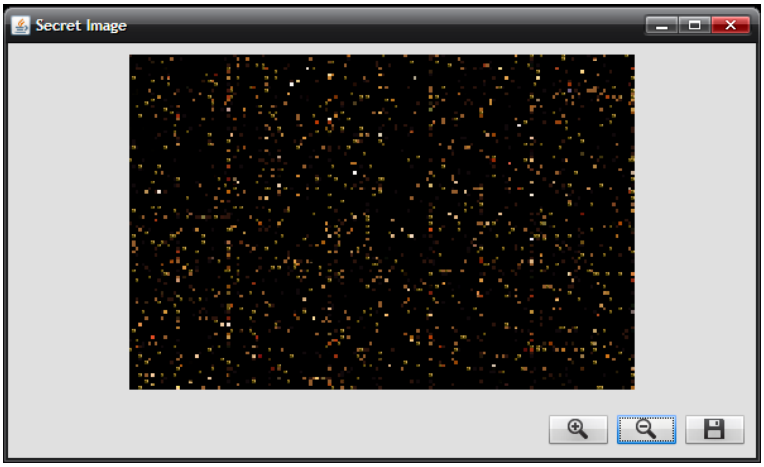


Figure 6.10 The recovered images of Figure 6.9 which are noise.

Chapter 7
Conclusions and Suggestions for Future Works

7.1 Conclusions

In this study, we have proposed two methods for creation of full-color and grayscale secret-fragment-visible mosaic images, respectively, and a data hiding technique has proposed for this type of mosaic image. Furthermore, by utilizing the creation process, we have proposed an image steganographic scheme by the use of grayscale secret-fragment-visible mosaic images, and presented a secret sharing method via secret-fragment-visible mosaic images.

For the full-color secret-fragment-visible mosaic image creation process, first, we have extended a technique of content-based image retrieval by proposing a new 1-D *h*-colorscale to represent the color distribution of an image more effectively according to human visual feeling of an image. With the 1-D *h*-colorscale, we have proposed next a new *h*-feature for image similarity measure computation, and use this value as the selection criterion of a greedy algorithm proposed for fitting a tile image into an appropriate target block using less computation time. To solve the problem of using an insufficiently-large database, we have proposed a remedy method by enlarging a selected target image in proportion to the error between the target image and the input secret image.

For the data hiding method which is used in covert communication via secret-fragment-visible mosaic images, we have made good use of the transformation

of the 1-D *h*-colorscale histogram. Tile images which are in the same histogram bin have the similar colors. By switching the relative positions of corresponding target blocks of such tile images, we can embed secret messages into a secret-fragment-visible mosaic image imperceptibly.

For use of grayscale secret-fragment-visible mosaic images for the steganographic purpose, we have utilized the distribution of grayscale values for image similarity measure computation. Furthermore, we have proposed a creation scheme to create a grayscale secret-fragment-visible mosaic image by a greedy algorithm which utilizes a new feature, called *k*-feature, to decrease the running time of fitting tile images into target blocks. With this method, users can keep or transmit the mosaic image instead of the secret document. This reduces the risk of revealing important documents to attackers or hackers.

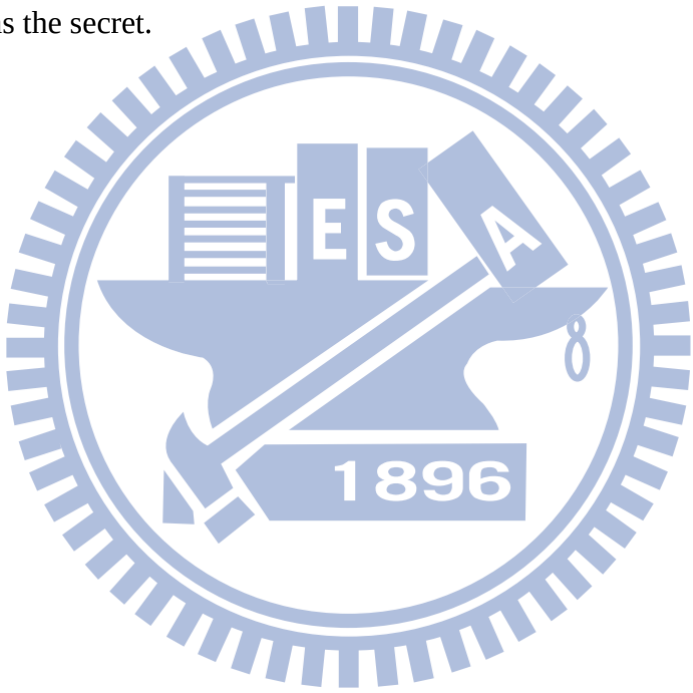
For secret sharing, we have proposed a method through the generation of multiple secret-fragment-visible mosaic images. With the 1-D *h*-colorscale histograms of the resulting mosaic images, we can assign labels to the selected target images, and generate a new recovery sequence for each of them based on the order of labels, resulting in a good property for secret sharing — the secret can be regained only if all the shares are all gathered together. Because each original recovery sequence is distributed in other mosaic images, the only way to retrieve the secret is to collect all the shares together and using the right key in the proposed secret recovery process.

7.2 Suggestions for Future Works

There are several interesting topics worth further study as listed in the following.

1. For the mosaic image creation process, it is interesting to allow a user to select his/her desired target image to create a secret-fragment-visible mosaic image.

2. It can be tried to apply a reversible color shifting technique to fit the color distribution of the secret image to a selected target image in order to achieve the above-mentioned goal.
3. It is worthwhile to develop visible watermarking techniques on the type of secret-fragment-visible mosaic image proposed in this study.
4. It is beneficial to extend the proposed methods to survive attacks such as scaling, rotation, JPEG compression, etc.
5. It is interesting to generalize the proposed method of steganography for use of the video as the secret.



References

[1] S. C. Hung, D. C. Wu, and W. H. Tsai, "Data hiding in stained glass images," *Proceedings of 2005 International Symposium on Intelligent Signal Processing and Communications Systems*, Hong Kong, June 2005, pp. 129-132.

[2] Gilberto Viciedo. (September 2009). Leonardo - Mona Lisa. [Online]. Available: http://fineartamerica.com/featured/leonardo--mona-lisa-gilberto-viciedo.html

[3] Around Hitchin. (November 2008). Mosaic Frieze. [Online]. Available: http://aroundhitchin.net/?p=212

[4] P. Haeberli, "Paint by numbers: abstract image representations," *Proceedings of 1990 International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 1990)*, Dallas, USA, 1990, pp. 207-214.

[5] A. Hausner, "Simulating decorative mosaics," *Proceedings of 2001 International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 01)*, Los Angeles, USA, August 2001, pp. 573-580.

[6] Y. Dobashi, T. Haga, H. Johan and T.Nishita, "A method for creating mosaic image using voronoi diagrams," *Proceedings of 2002 European Association for Computer Graphics (Eurographics 02)*, Saarbrucken, Germany, September 2002, pp. 341-348.

[7] J. Kim and F. Pellacini, "Jigsaw image mosaics," *Proceedings of 2002 International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 02)*, San Antonio, USA, July 2002, pp. 657-664.

[8] G. D. Blasi, G. Gallo, and M. Petralia, "Puzzle image mosaic", *Proceedings of 2005 International Association of Science and Technology for Development on Visualization, Imaging and Image Processing (IASTED/VIIP 2005)*, Benidorm,

Spain, September 2005.

[9] D. C. Wu and W. H. Tsai, “Embedding of any type of data in images based on a human visual model and multiple-based number conversion,” *Pattern Recognition Letters*, vol. 20, pp. 1511-1517, August 1999.

[10] J. Fridrich, M. Goljan, and R.Du, “Lossless data embedding—new paradigm in digital watermarking,” *European Association for Signal Processing (EURASIP) J. Appl. Signal Processing*, vol. 2002, no. 2, pp. 185-196, 2002.

[11] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, “Lossless generalized LSB data embedding,” *IEEE Transactions on Image Processing*, vol. 14, no. 2, pp. 253-266, February 2005.

[12] J. Tain, “Reversible data embedding using a difference expansion,” *IEEE Transactions on Circuits System Video Technology*, vol. 13, no. 8, pp. 890-896, August 2003.

[13] A. M. Alattar, “Reversible watermark using the difference expansion of a generalized integer transform,” *IEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 97-105, March 2003.

[14] D. Coltuc and JM. Chassery, “Very fast watermarking by reversible contrast mapping,” *IEEE Signal Processing Letters*, vol. 14, no. 4, pp. 255-258, April 2007.

[15] W. L. Lin and W. H. Tsai, “Data hiding in image mosaics by visible boundary regions and its copyright protection application against print-and-scan attacks,” *Proceedings of 2004 International Computer Symposium (ICS 2004)*, Taipei, Taiwan, Republic of China, December 15-17, 2004.

[16] C. C. Wang and W. H. Tsai, "Creation of Tile-overlapping mosaic images for information hiding," *Proceedings of 2007 National Computer Symposium*, Taichung, Taiwan, Republic of China, December 20- 21, 2007, pp. 119-126.

[17] C. Y. Hsu and W. H. Tsai, “Creation of a new type of image - circular dotted image - for data hiding by a dot overlapping scheme,” *Proceedings of 2006 Conference on Computer Vision, Graphics and Image Processing*, Taoyuan, Taiwan, Republic of China, August 13-15, 2006.

[18] C. P. Chang and W. H. Tsai, “Creation of a new type of art image—tetromino-based mosaic image—and protection of its copyright by losslessly-removable visible watermarking”, *Proceedings of 2009 National Computer Symposium*, Taipei, Taiwan, Republic of China, November 27-28, 2009, pp. 577-586.

[19] J. R. Smith and S. F. Chang, “Tools and techniques for color image retrieval,” *Proceedings of Society for Imaging Science and Technology and SPIE (IS & T/SPIE)*, vol. 2670, February 1995, pp. 2-7.